

ג'בראון

למתחילים



LIST

```

10 PMODE=1:SCREEN1,1:PCLS
20 FORR=15TO60STEP10
30 CIRCLE(128,96+R),R,1,.5,1:CIRCLE(
R,1,0,.5:CIRCLE(
128-R,96),R,1,.75,.25:CIRCLE(12
8+R,96),R,1,.25,.75
79 NEXTR
90 FORR=80TO20STEP-15
100 CIRCLE(128,96),R:NEXTR
1000 GOTO1000
OK

```



דף זה נותר ריק במכוון.

בסיק

למתחילים



BASIC

לחובב, לסטודנט או למתחיל. למדו את
שפת ה-BASIC החדשה בגירסת
ALTAIR. המותאמת למיקרו ולמיני
מחשבים

מאת
ג'רלד בראון



INSTANT BASIC

FREEZE-DRIED COMPUTER PROGRAMMING IN
BY
JERALD R. BROWN

מאנגלית: פאני מאוריבר

עיצוב העטיפה: שלמה ניאגו

© Copyright, 1982, dilithium Press

©
כל הזכויות שמורות
פרסום בכל צורה שהיא, שלם או חלקי מספר זה
אסור ללא רשות בכתב מההוצאה.

נדפס בישראל
סודר במסדרה האלקטרונית של הוצאת "אור-עם"
מהדורה ראשונה - תשמ"ג/1983
83 84 85 86 87 88 89 - 9 8 7 6 5 4 3 2 1

2	פרק א: מוכנים היכון רוץ
20	פרק ב: תאים קטנים
73	פרק ג: לולאות לולאות, אוב מלים אחרות GOTO
46	פרק ד: משתנים, נקודה צה וחסכון בעבודה
54	פרק ה: השווה והחלט משפחת ה-IF...THEN
66	פרק ו: מסעף פונקציות - 1
82	פרק ז: לולאות אוטמאטיות
96	פרק ח: מסעף פונקציות - 2
117	פרק ט: מלכות המשתנים המצוינים
140	פרק י: מנע אותם - 2

BUG

MICROCOMPUTERS BOOKS & SOFTWARE

מרכז לספרות עזר ותוכנה למחשבים

רחוב גוף סנטר (שער 5) ת"א
בניין כלל חנות 260 ירושלים

תוכן

תוכנן והופק עלידי המחבר בחברת דיימקס. עריכה, עידוד
וסיכום הבעיות עלידי לרוי פינקל. הערות והצעות עלידי
בוב אלברכט הדגול. הדפסה וניהול עלידי מארי ג'ו מקפי.
עיצוב העטיפה ויצירות האמנות בעמודים 3, 4, 8, 48 ו-128
על עלידי אן מיא. תודתי לכל אלה שסבלו בעת סיום כתב
היד הראשון וסייעו בגיבוש ההסברים והתרגילים.

*ALTAIR is the registered trademark of MITS
DEC BASIC PLUS is the registered trademark
of Digital Equipment Corporation
BASIC is the registered trademark of Dartmouth
College*

יש ספרים רבים שנכתבו במטרה ללמד ולהקנות את שפת התכנות BASIC. ספרנו זה הוא הנהדר מכולם בזכות היתרונות הבאים:

- * יעיל
- * קל להבנה
- * חסכוני
- * מותאם למיקרומחשבים ולמיני-מחשבים
- * בלי מתמטיקה מסובכת
- * סיכומים מסודרים של BASIC לאורך כל הספר
- * בחנים ותרגילים שיעידו עד כמה היטבנו ללמדך את שפת ה-BASIC

אנו מנצלים במלואה את תכונת ההיזון החוזר המידי של ה-BASIC, כדי להציע לכם המחשות מעשיות ותרגילים ללימוד מהיר וקל. בתוך 6 עד 16 שעות תתוודעו אל ה-BASIC מקרוב ואף תוכלו לעבוד בשפה זו. בדוגמאות שלנו, אנו משתדלים לצמצם את זמן ההדפסה למינימום, בעוד שאנו מקדישים את מירב הזמן להקניית ולפיתוח שיטות תכנות נכונות. הניסיון שצברנו בהוראת ה-BASIC למתחילים שגילם נע בין 6 ל-60 מלמד, שבתכנות קיימים שלושה מושגים העלולים להקשות על מתחילים: ההפנייה המותנית (IF...THEN), לולאות או חזרות (FOR...NEXT), מיון (משתנים מאופיינים). נקדיש תשומת לב מיוחדת למושגים אלה ונמחישם בדוגמאות שעובדנו בעצמנו ושבמשך השנים הוכיחו עצמן כיעילות ביותר. אנו יוצאים מנקודת הנחה שאתם מתחילים וחסרי ניסיון בשטח התכנות. אנו מניחים שיש לכם גישה נוחה למחשב או למערכת BASIC PLUS. בעלי הניסיון מבינים, ואלה מכם שעבדו לפי אחת הגירסאות הישנות של BASIC יוכלו לנשום לרווחה בפרקים הראשונים.

חפש את הסיכומים המופיעים במסגרת נקודות,
והמפוזרים בספר כולו.

במחציתו הראשונה של הספר, אנחנו מתקדמים לאט לאט. אנו מעודדים אתכם (בעצם אנו תובעים הרבה מאוד) לנסות לבטא את כישרונכם בתחום התכנות, מעבר לדוגמאות ולתכניות שאנו כוללים בספרנו. אנו רוצים שתחשבו על הדברים המעניינים והמסקרנים אתכם ותנסו ליישם את הנלמד כאן לאותם דברים.

ספר זה, המקדיש תשומת לב רבה לעבודה עצמית, מהווה תחליף לספרים אחרים בנושא, המציעים תשובות ופיתרונות לבעיות.

כל התכניות וההרצות המופיעות בספר בוצעו בגירסת Altair 8K BASIC 3.2, הוזה ל-BASIC PLUS. ההבדלים בין שתי גירסאות ה-BASIC מאופיינים שני ניבים של אותה שפת BASIC ומוזכרים בסיכומים שבתוך מסגרת הנקודות.



פירוש הדבר שאנו רוצים רק שתקרא את מה שבהמשך

הבה נתחיל, איפוא. האם המחשב פתוח? האם המקלדת פתוחה? אם לא, מצא מישהו שיראה לך כיצד מפעילים אותה. בנוסף לכך, אם אתה חדש בשטח זה, יהיה על מישהו ללמדך את הדברים הבאים:

- (1) האם המכונה (המחשב) פתוחה.
- (2) האם הקוד שלך הודפס והאם המחשב מוכן להיענות לך.
- (3) האם ה-BASIC מוכן לשימוש במחשב שלך? אם לא, מישהו יצטרך להסביר לך כיצד לטעון את המחשב ב-BASIC. אם הינך בעל ניסיון וגם סבלני בנוסף לכך, אתה עשוי להצליח בהטענת ה-BASIC במיקור מחשב שלך בעזרת חוברת ההוראות המצורפת למחשב.



המכשיר או הכלי שבאמצעותו תעביר את הוראותיך למחשב ותקלוט את תשובותיו הינו דמוי מכונת כתיבה ובעל מקלדת. היא אמנם אינה זהה לזו של מכונת כתיבה, אך דומה לה מאוד. היא קרויה מסוף. זהו מונח נוסף הלקוח מעולם המחשבים והחייב להתחיל להיות שגור בפיך.

המקלדת היא אמצעי קלט/פלט, והיא אחד ההתקנים הנלווים למחשב. המחשב עצמו, שבו מתבצעות פעולות החישוב השונות, מכונה יחידת העיבוד המרכזית (CPU).





פירוש הדבר שאנו רוצים שתנסה זאת במסוף המחשב שלך; קדימה, עשה זאת.

האם המכונה שלך מופעלת ופועלת בשפת ה-BASIC? כדי לקבל את התחושה הנכונה, נסה להדפיס HELLO COMPUTER

לאחר שסיימת להדפיס, לחץ על קליד ההחזרה RETURN, הנושא לעתים את התווית "CR".



מקלדת

המחשב ידפיס בתגובה את המסר הבא:

HELLO COMPUTER ←

אתה מדפיס זאת ולוחץ על RETURN.

?SN ERROR
OK

המחשב אומר לך שעשית טעות; כל מה שהוא אינו מבין נחשב על-ידו לטעות.

SN ERROR - הוא הסימן לשגיאת תחביר וזו דרכו של המחשב לומר, "בסדר, עשה זאת נכון, ראש כרוב שכמוך!"



הבעיה שהמחשב לא הבין כלל מה הדפסת. עדיין לא המציאו מחשבים שיוכלו להבין באנגלית פשוטה מה רוצים מהם. וכך, כדי להביא את המחשב לידי כך שיעשה משהו, אנו משתמשים בשפת מחשב, כדוגמת BASIC, כדי להציג את דרישותינו למחשב.



היכון...



B.A.S.I.C.

שפת המחשב הזו (BASIC) היא פשוטה מאוד. שפת ה-BASIC הומצאה בדארטמות ושמה מהווה ראשי תיבות של:

Beginners All-purpose Symbolic Instruction Code.

היא נועדה להקל עלינו, הלא מקצוענים, את תכנות המחשבים. ייתכן שכבר שמעתם על שפות מחשב אחרות, כגון: פורטרן, קובול, אפל ורבות אחרות. כל שפה מורה למחשב לבצע דברים, אך הן נועדו להקל על שימושים שונים בו. ה-BASIC היא שפה כללית ללא-מקצוענים, בעוד ה-COBOL לדוגמא, מיועדת לשימושים הקשורים לעולם העסקים והיא משמשת מתכנתים מקצועיים.

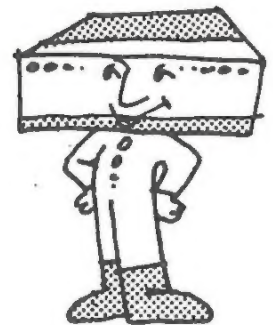
ה-BASIC משתמשת במלים באנגלית להעברת הוראות למחשב. למרות ששפת ה-BASIC מורכבת ממלים וסימנים מועטים יחסית, יש להציג הוראות אלו למחשב לפי דפוס נוקשה ומדויק, הלא הוא תחביר השפה. זה מה שאנחנו מתחילים ללמוד עכשיו.

ברוב המקלדות הקיימות במסופי מחשב, נדפסות אותיות גדולות, CAPITALS, מבלי שנלחץ על הקליד SHIFT. המסוף שלכם יציג את התוצאה על מסך וידאו או ידפיסה על גבי נייר. הדפס את המלה NEW (כולה באותיות גדולות) ולחץ על קליד ההחזרה RETURN. המחשב יענה ב-OK או במסר דומה.



OK

NEW הדפס NEW ולחץ על RETURN.
המחשב הדפיס זאת ועתה הוא
OK מחכה בסבלנות לצעד הבא שתעשה.



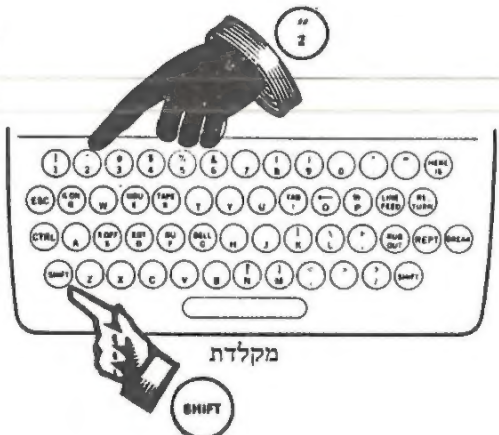
כל ההוראות שהיו קודם לכן בזכרוננו של המחשב נמחקו, והוא מוכן לקבלת הוראות חדשות. הדפס, איפוא, את ההוראה החדשה:

20 PRINT "THIS IS EASY"

ואז לחץ על הקליד RETURN. להלן מספר רמזים מחכימים:

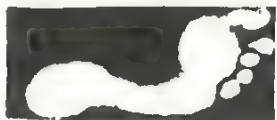
כדי לקבל מרכאות, לחץ על קליד SHIFT ואל תשחרר כשאתה לוחץ על "2"

אם טעית, לחץ על RETURN והתחל שנית. בעמודים הבאים תתוודע לשיטות יעילות יותר להכנסת תיקונים.



” ”

הרץ אותה...



הדפס עתה את המלה RUN ולחץ על הקליד RETURN.

אתה מדפיס RUN ולוחץ על קליד * * * * * ← RUN
THIS IS EASY ← RETURN

OK ← המחשב מדפיס (PRINTs) זאת ומשיב OK, כשהכוונה היא "אני אכן מריץ (RUNning) את תכניתך."



ברכותיגו! זה עתה הכנסת או הדפסת בתכנית מחשב (תכנית קטנה, המורכבת משורת הוראות יחידה), וגרמת למחשב לרוץ (RUN) ולבצע את תכניתך. למעשה, הורית למחשב להדפיס (PRINT) את מה שבתוך המרכאות. הוראתך או תכניתך נראתה כך:

20 PRINT "THIS IS EASY"

המחשב השיב בהדפיסו

THIS IS EASY

שלאחריו OK לאישור הביצוע.



המספר הסידורי של השורה



בואו נשתלט על עוד מספר מונחים ועל כמה מן החוקים המרכיבים את שפת המחשב, הקרויה BASIC. כשאתה אומר למחשב להריץ (RUN) תכנית, הוא מתחיל בביצוע ההוראה בעלת המספר הסידורי הנמוך ביותר. ההוראה שנתת למחשב מכונה הוראת ה-PRINT. כשהדפסת RUN ולחצת על RETURN השיב המחשב בכך שהדפיס את המידע שהיה מצוי בין המרכאות. מה שבין המרכאות מכונה מחרוזת.



“
”
STRING

בתכנית BASIC, הוראה כדוגמת:

20 PRINT "THIS IS EASY"

מתחילה תמיד במספר סידורי, היכול להיות כל מספר חיובי מ-1 עד 65529. בדוגמה שלנו השתמשנו ב-20 כמספר סידורי.

הוראות הפעלה כדוגמת NEW ו-RUN אינן מהוות חלק מתכנית ולכן אינן מתחילות במספר סידורי. הואיל וספר זה נועד ללמד אותך את ההוראות התכנות, נתרכז בהוראות המתחילות במספר סידורי. כשנדבר על סתם "הוראה" נתכוון להוראות המרכיבות את תכנית המחשב ולא להוראות ההפעלה.

כשהמחשב מסיים לעשות או "לכצע" הוראה, הוא ממשיך בביצוע ההוראה הבאה, לפי הסדר המספרי. כלומר, ההוראה בעלת המספר הסידורי הגבוה יותר, היא ההוראה הבאה שיש לבצע. כשהינך מריץ תכנית, אנו אומרים שהמחשב מבצע את ההוראות. פירושו של דבר שהוא פועל, אך לא רק עם מספרים.

תולעת הספרים הקטנה
שמתחפרת בתוך
הספר אומרת...



PRINT



Print



בצע

קודם כל הדפס NEW ולחץ על RETURN. ואז הדפס את התכנית הבאה.

NEW

OK

10 PRINT "WHAT"
20 PRINT "A"
30 PRINT "BREEZE"



לחץ על RETURN בסופה של כל
הוראה הכלולה בתכנית, וגם לאחר
שאתה מדפיס RUN.

RUN

WHAT

A

BREEZE

OK

המחשב מבצע קודם כל את השורה 10 ומדפיס
WHAT
ואז הוא עובר להוראה בעלת המספר הסיידורי
הגבוה יותר, שורה 20, ומדפיס A. לאחר שורה 20
מבצע המחשב את ההוראה שבשורה בעלת המספר
הסיידורי הגבוה יותר, שורה 30. אל תספר לי, הנח לי
לנחש מהו מדפיס.

ועתה ערוך תכנית דומה לזו, כשאתה משתמש בהוראת ה-PRINT לצורך
הדפסת מחרוזת אחת או יותר. אל תשכח את המרכאות שלפני ואחרי המחרוזת.

NEW

OK

הבא את המחשב
שלך לידי כך
שישיב לך!



קרא

אינך חייב להכניס או להדפיס את התכנית לפי סדר המספרים. כלומר,
אינך חייב להדפיס תחילה את שורה 10, לאחריה שורה 20 ולבסוף שורה 30.
אם אנו מכניסים תכנית שלא לפי סדר המספרים, אין זה מפריע כלל וכלל
למחשב. הוא מבצע את ההוראות לפי המספרים הסיידוריים ולא לפי סדר
הכנסתם לתכנית או הדפסתם. כך קל יותר להכניס שורות נוספות לתכנית,
שכבר מצוייה בזיכרונו של המחשב. ייתכן שכבר הבחנת שאנו נוהגים למספר
את השורות העוקבות בתכנית במספרי עשרות שלמות. הדבר מקל על הוספת
הוראות נוספות, בין מספרי השורות הקיימים - עד 9 הוראות נוספות בין
השורות 10 ו-20, לדוגמא.

המחשב מגלה את צפונותיו



ועתה הדפס NEW, לחץ על RETURN והכנס את התכנית שלהלן: בדוק את המספרים הסידוריים.

NEW

```
OK
30 PRINT "BABY"
10 PRINT "TO"
20 PRINT "ME"
RUN
TO
ME
BABY
OK
```

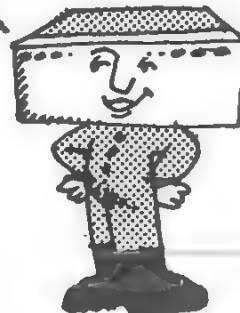
בראש וראשונה הדפסנו שורה מספר 30 ולחצנו RETURN. ואז הדפסנו שורות 10 ו-20. המחשב ערשה חיל.

עתה הדפס LIST ולחץ על RETURN. אתה רואה כיצד ה-BASIC סידרה מחדש את ההוראות בתכנית לפי סדר המספרים הסידוריים?

LIST

```
10 PRINT "TO"
20 PRINT "ME"
30 PRINT "BABY"
OK
```

LIST



LIST מורה למחשב לערוך ולסדר את התכנית שהדפסת זה עתה. הוראת ה-LIST תסדר את ההוראות שבתכנית בהתאם למספר הסידורי. אתה יכול לסדר תכנית כל זמן שאינך מריץ תכנית, אם תרצה לדעת מה אצור בזיכרוננו של המחשב, הדפס LIST ולחץ על RETURN והמחשב יגלה לך את כל צפונותיו.

קרא

הבה נשתעשע מעט עם הוראת ה-PRINT כך, שתקבל כמה רעיונות
באשר לשימושיה השונים. ראשית, הדפס NEW והדפס RETURN. המחשב יתן
לך אור ירוק באמצעות OK.

NEW

אתה מדפיס NEW ולוחץ RET

OK

BASIC מוכנה לצאת לעבודה למענך

השתמש ב- (=) לסימן מינוס

השתמש ב- (SHIFT) ו- (+) לסימן פלוס.

ואז הדפס את מה שבהמשך:

10 PRINT 12+12

20 PRINT 12-12

הדפס בזהירות
שתי פקודות
תכנית אלן.

ועוד מספר רמזים -

- (1) עליך ללחוץ על RETURN לאחר שסיימת להדפיס
הוראה ולפני שתדפיס את המספר הסידורי הבא.
- (2) באותיות הקטנות, אין אפשרות להדפיס את האות
L במקום המספר (1) ואת האות O במקום אפס.



מקלדת

כשהינך מריץ את התכנית דלעיל, ידפיס המחשב כך את התשובות
לבעיות:

RUN

אתה מדפיס RUN ולוחץ על RETURN

24

0

המחשב עורך את החישוב ונותן את התשובות
קיימות 2 תשובות, לכל פקודת תכנית.

OK

"הכול בוצע, בוס!"



או כמו שאומרים במחשבת (או בשפת המחשב), המחשב מחשב את ערך
הביטוי (12 + 12 ו-12 - 12), ולפי ההוראות מדפיס את התשובות.

להלן מידע נוסף על חישובי BASIC:

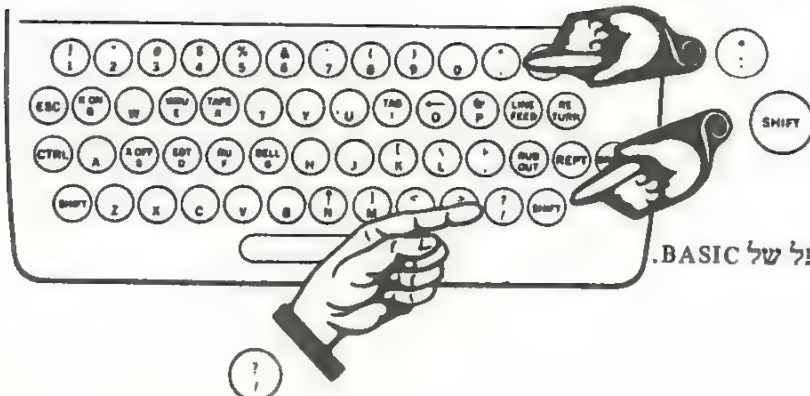
- * כדי לומר למחשב לחבר, השתמש בסימן +
- * כדי לומר למחשב לחסר, השתמש בסימן -
- * כדי לומר למחשב להכפיל, השתמש בסימן *
- * כדי לומר למחשב לחלק, השתמש בסימן /

+

-

*

/

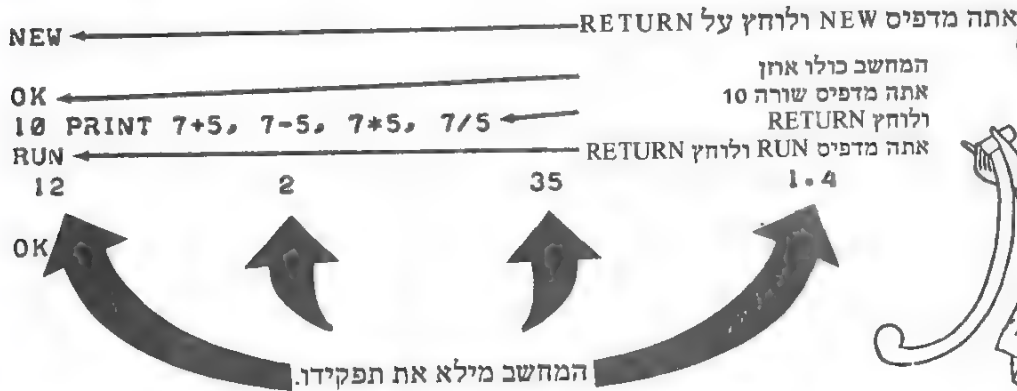


השתמש ב- (SHIFT) ביחד עם (:) כסימן הכפל של BASIC.

השתמש ב- (SHIFT) כסימן החילוק.



ראשית חכמה, מחק את תכניתך האחרונה מזכרונו של המחשב באמצעות הדפסת NEW. וכעת, הדפס תכנית זו והרץ אותה.



מהרצת תכנית זו תוכל ללמוד כיצד מפרידים הפסיקים בין החלקים השונים של הוראת ה-PRINT. שים לב שההוצאות שנתן המחשב בעקבות הוראת ה-PRINT שלך מפורזות על פני השורה.

כשהוראת PRINT מכילה יותר מחלק אחד, יש להפריד בין החלקים השונים על ידי סימן פסיק או נקודה פסיק.



נסה זאת.

NEW

OK

10 PRINT 1*1, 2*2, 3*3, 4*4, 5*5, 6*6, 7*7, 8*8, 9*9

RUN

1	4	9	16	25
36	49	64	81	

OK

PRINT



כמו שהינך רואה, השימוש בפסיק להפרדת החלקים השונים בהוראת PRINT אחת, מקצה לך חמישה קטעים לאורך השורה. בשורה יש 72 רווחים או תווים. (תו הוא כל מספר, אות, סימן או רווח.) אם אתה טיפוס הססן, הדפס סימני \times לאורך השורה והמשך לספור.

XX

יש מסופים בהם יש יותר מ-72 תווים בשורה, ומסופים אחרים (בעיקר בעלי מסך וידאו) שיש להם פחות מ-72 תווים בשורה. הואיל והשורות בעלות 72 תווים הינן הנפוצות ביותר, נאמץ אותן לעצמנו בספרנו זה. אתה להוט לשמוע פרטים? בבקשה. הפסיק, בהוראת ה-PRINT, מחלק את השורה ל-5 קטעים, שבכל אחד 14 תווים. שני התווים האחרונים בכל שורה אינם בשימוש.

דרך אגב, ה-BASIC מונה את התו הראשון כאפס והאחרון כ-71 במניין, ולא מ-1 עד 72. (פרטים נוספים בהמשך.) בשורה שלהלן קרא את המספרים מלמעלה למטה, כך: $\frac{7}{1} = 71$ $\frac{1}{0} = 10$

0123456789111111112222222222333333333344444444445555555555666666666677
01234567890123456789012345678901234567890123456789012345678901



PRINT

הדפס את הערכים במסוף.
(מספר סידורי) PRINT

```

20 PRINT 7 + 3
30 PRINT "INSTANT BASIC"
40 PRINT N$
    
```

מהו השימוש? כדרך מקוצרת להכנסת PRINT, סימן הפסיק מחלק את השורה ל-5 קטעים. הנקודה פסיק גורמת לכך שהערכים יודפסו זה לצד זה.

```

10 PRINT X,Y,Z$
20 PRINT A;B;C
    
```

פסיק או נקודה פסיק בסופה של הוראת PRINT מבטלים את החזרת המרכב.



ועתה הכנס (הדפס) את שתי ההוראות שלהלן. שים לב שבשורה 20 אנו משתמשים בסימני נקודה פסיק במקום בפסיקים כדי להפריד בין החלקים השונים.

NEW

OK

10 PRINT 7+5, 7-5, 7*5, 7/5

20 PRINT 7+5; 7-5; 7*5; 7/5

RUN

12 2 35 1.4 35

1.4 ← הודפס ע"י שורה 10 בה השתמשנו בפסיקים.

הודפס ע"י שורה 20 בה השתמשנו בסימני נקודה פסיק.

OK



הסתכל בתוצאות שקיבלת מהרצת התכנית. תן דעתך לצורה בה מדפיס המחשב את התשובות לארבע הבעיות, כשהן מופרדות עלידי פסיקים (שורה 10).

10 PRINT 7+5, 7-5, 7*5, 7/5

בהשוואה לתשובות לארבע הבעיות המופרדות עלידי סימני נקודה פסיק.

20 PRINT 7+5; 7-5; 7*5; 7/5



השתמש בנקודה פסיק במקום בפסיק כדי להורות למחשב לצופף את הפלט. אתה רשאי להשתמש ב-; וב-, בהוראת PRINT אחת.



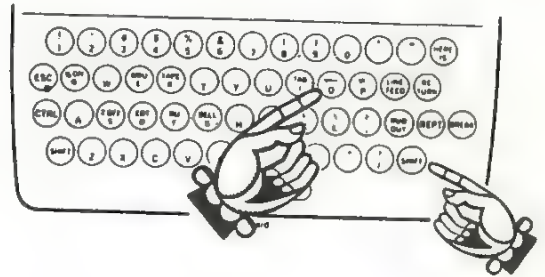
S שגיאים

האם אתה שוגג לפעמים? אנחנו כן. הבט.

שגינו בכתיב של המלה PRINT 2*3+4
RUN

המחשב אומר לנו ששגינו. בדיקה בחוברת ההוראות
מראה SN ERROR ב-10 שפירושה שגיאת 10 SN ERROR
תחביר. OK

יש להדגיש שאילו הבחנו בכך שהקשנו על T בעוד שהתכוונו להקיש R, יכולנו לתקן את טעותנו עלידי הקשה על (O). כדי להדפיס זאת עליך להקיש על הקליד (SHIFT) (O), ולהחזיק ובו זמנית להקיש על (O).



לחץ על (SHIFT) ו- (O) ביחד.



NEW

OK

10 PT-RINT 2*3+4

LIST

הקליד (O) מבטל את הסימן שעליו הוא מצביע.

LIST (מסדר את התכנית)

10 PRINT 2*3+4

OK

RUN

10

אתה רואה? ההוראה תקינה עתה.

OK

NEW

OK

10 PRINT "GET H-THE POIM --NT"

מוחק, משמיט את ה-H

מבטל את הרווחים ואת ה-M

LIST

10 PRINT "GET THE POINT"

OK

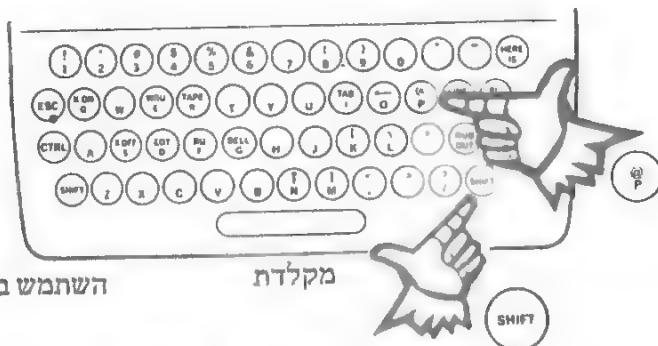


עוד שגיאים בעמוד הבא...

האם חשבת אי פעם ללמוד
הדפסה במכונה בשיטה
עיוורת?



אם אתה מדפיס הוראה... ולפתע פתאום... אתה רוצה לבטל את כל השורה שהדפסת, שפת ה-BASIC אינה מרשה לך להתחיל מחדש. אתה פשוט מדפיס @ בעודך מקיש על הקליד SHIFT.



השתמש ב- SHIFT - וב- P - בו זמנית.

מקלדת



10 PRINT "GET THE OIUB@"

נבהלנו והחלטנו לבטל את השורה כולה. המחשב מבצע החזרת מרכב, בדיוק כאילו הקשתם על RETURN.



נניח שהינך מעוניין לבטל שורה מתכנית שכבר הספקת להכניס למחשב, וזאת מבלי לפגוע בכל השאר. אינך רוצה להתחיל מחדש ולהשתמש ב-NEW. כל שעליך לעשות הוא להדפיס את מספרה הסידורי של השורה שאתה רוצה לבטל ואז להקיש על RETURN. ראשית, אנו מביאים כאן תכנית קטנה.

NEW

OK
10 PRINT "NOT"
20 PRINT "VERY"
30 PRINT "MUCH"
RUN
NOT
VERY
MUCH

OK

אל תשכח את
המרכאות.



המשך יבוא....



ועתה אנו רוצים לבטל שורה 10, כך שהתכנית תדפיס רק VERY MUCH.
אל נא תדפיס NEW. עבוד לפי ההוראות שבהמשך.

```

10 ← הדפס את המספר והקש על RETURN
LIST ← הדפס LIST ו-RETURN
20 PRINT "VERY" ← שורה 10 נעלמה לעולם, או עד שתכניסנה מחדש.
30 PRINT "MUCH"
OK
RUN ← הרץ זאת
VERY
MUCH
OK

```



אתה רשאי גם להחליף שורה בתוך התכנית. כדי להחליף הוראה, עליך להדפיס את מספר השורה של ההוראה שהינך מעוניין להחליף ולהמשיך בהדפסת ההוראה הבאה. כשהינך מסיים את ההוראה ומקיף RETURN, כבר החליפה השורה החדשה את ההוראה הישנה.



LIST

```

20 PRINT "VERY" ← אנו רוצים להחליף שורה זו.
30 PRINT "MUCH" ← (שתי ההוראות נמצאות עדיין בזיכרונו של המחשב.)
OK

20 PRINT "TOO" ← הדפס שורה זו עם מספר השורה
LIST ← אותה הינך רוצה להחליף.
      ← כדי לראות את התכנית החדשה
20 PRINT "TOO" ← התכנית החדשה שלנו היא הרבה יותר...
30 PRINT "MUCH"
OK
RUN
TOO
MUCH
OK.

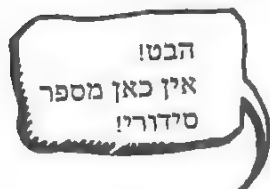
```



גישה ישירה

(direct mode)

בשפת ה-BASIC בסגנון ALTAIR קיימת גישה ישירה, המאפשרת את השימוש בהוראת PRINT לצורך חישוב ולצורך הדפסת התוצאות. אתה יכול להשתמש בגישה ישירה כדי לערוך ניסויים שונים עם האריתמטיקה של BASIC. במקום להשתמש במספר סידורי, כנהוג בהכנסת הוראת PRINT, אתה מדפיס ישירות.



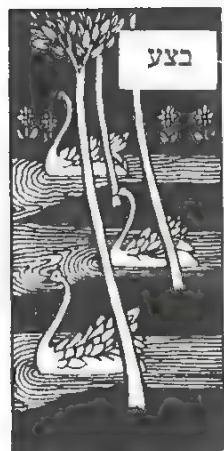
PRINT 3*5, 3+5, 3-5, 3/5



אתה יכול לתקן טעויות בגישה ישירה, בדיוק כפי שהיית עושה במקרה של הוראה רגילה.

בזמנך תלמד דברים נוספים שניתן לעשות בגישה ישירה. לעת עתה אתה רשאי להכניס ישירות תכניות כנות שורה אחת ולקבל את הפלט בהקשה על כפתור (קליד ה-RETURN).

הנה כמה תכניות קצרות שיאפשרו לך להתוודע טוב יותר אל הוראת ה-PRINT ואל האריתמטיקה של שפת ה-BASIC. נסה אותם, בבקשה. (בוא נתערב שהאמנת שהשלב של אריתמטיקה בסיסית כבר נמצא מאחוריך).



NEW

OK

10 PRINT "8 + 2 = "; 8+2

RUN

8 + 2 = 10

OK

10 PRINT "8 + 2 = "; 8+2

””

עם

זוהי מחרוזת המצוייה בתוך מרכאות. המחשב אינו למתייחס להוראות האריתמטיות שבתוך מרכאות.

””

בלי

המחשב יבצע את פעולת החישוב הזו מפני שאינה מצוייה בתוך מרכאות ועל כן אינה מהווה מחרוזת.

סימן הנקודה פסיק מורה להדפיס את חלקי הוראת ה-PRINT בצפיפות.

כיצד פועלת שפת ה-BASIC



הכנס והרץ את התכניות הבאות: או אם אתה מעדיף, השתמש בגישה ישירה לפתור כל שורה או כל בעיה אריתמטית.

NEW

OK

10 PRINT 2*3+4, 2*3+4*5, 2*3/4

20 PRINT 2*(3+4), (2+3)*(4+5), (2+3)/(4+5)

RUN

10

26

1.5

14

45

.555556

OK

NEW

OK

10 PRINT 2*2*2*2*2

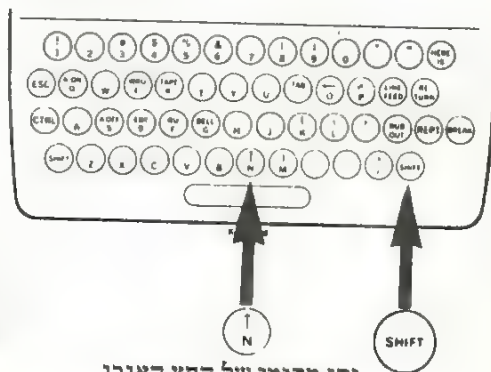
20 PRINT 2+5

RUN

32 2 x 2 x 2 x 2 x 2 is the same as 2⁵

32 (שניים בחזקת 5. שים לב לשימוש בחץ האנכי ב-BASIC לסימון העלאה בחזקה.

OK



זהו מקומר של החץ האנכי



החוקים

1. שפת ה-BASIC מחשבת ביטוי (לא בהכרח ביטוי חשבוני) לפי הסדר משמאל לימין.

2. ... עורכת בראש וראשונה את כל חישובי החזקות (+).....

3. ... ואז מתחילה שוב בצד שמאל ועוברת ימינה, כשהיא מבצעת את כל פעולות הכפל (*) והחילוק (/).....

4. ... ואז כשהיא שוב מתחילה בצד שמאל, מבצעת ה-BASIC את כל פעולות החיבור (+) והחסור (-).

5. שפת ה-BASIC מחשבת קודם כול את הביטויים שבתוך הסוגריים וזאת על פי סדר העדיפויות שהוגדר בסעיפים הקודמים.

6. אם יש זוגות סוגריים בתוך סוגריים, ההערכה או החישוב מתבצעים קודם כול בתוך הצמד הפנימי ביותר של סוגריים, אחר כך בזוג השני וכו'.

7. אך אל נא תשכח. לכל חלק שמאלי של סוגריים חייב להיות חלק ימני תואם ולהיפך, או ששפת ה-BASIC תודיע לך ששגית, בעת שתנסה להריץ את תכניתך

מובן מאליו שאתה יכול לפקח על הסדר שבו מבוצעים חישובי העלאה בחזקה או חישובים אחרים באמצעות השימוש בסוגריים - ראה 5, 6, 7 לעיל. הצטרף לתנועת הסוגריים, רבת העצמה.

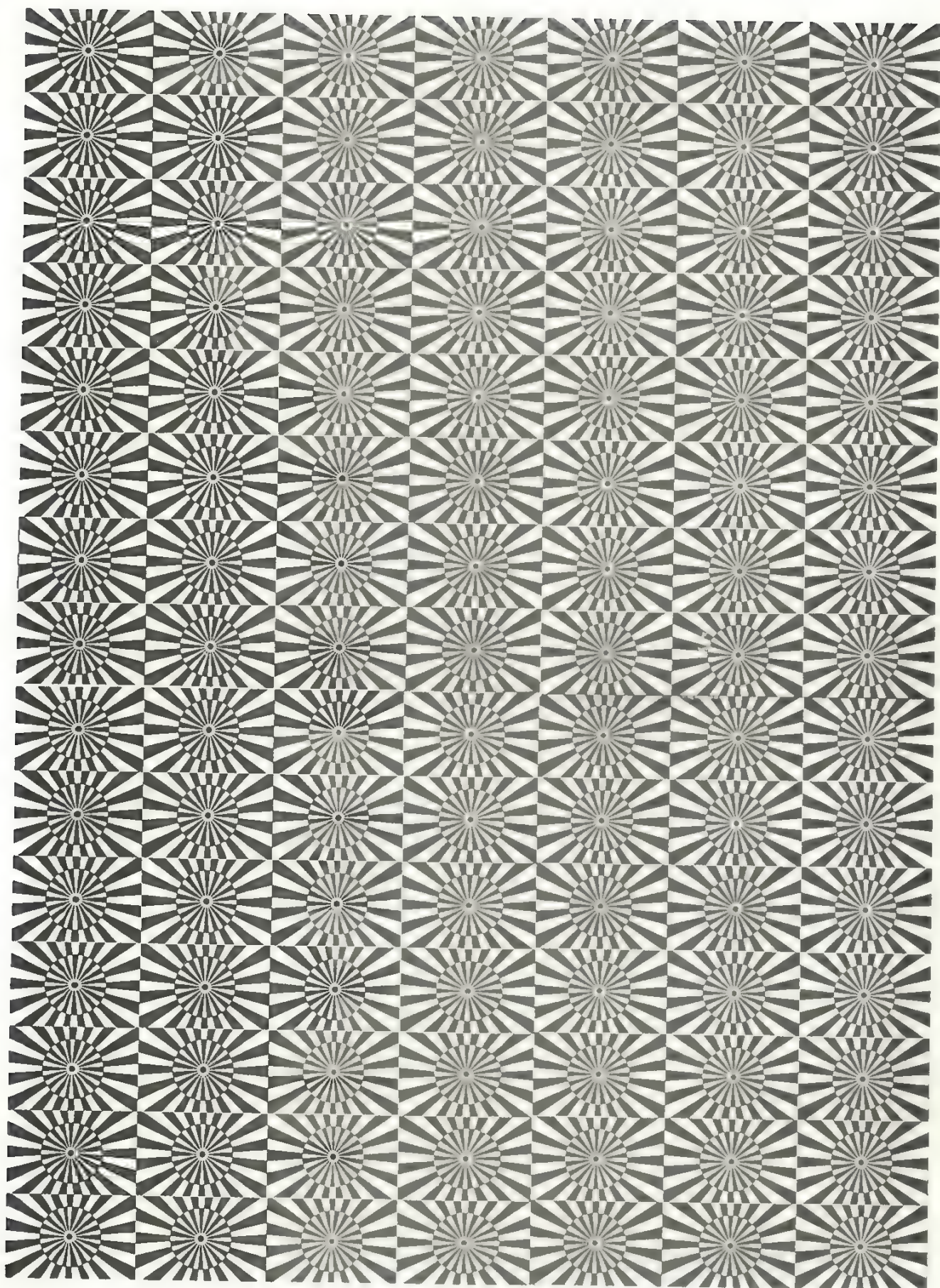
תוכנה אנושית



פרק 1: בעיות

1. חבר תכנית BASIC אשר:
 (א) הדפס את שמך,
 (ב) הדפס את כתובתך,
 (ג) הדפס את שם עירך והמיקוד שלך, בשלוש שורות עוקבות.
2. כתוב מחדש את תכנית 1, כאמצעות הוראה אחת שתרכז את כל המידע בשורה אחת שתופיע על מסך המסוף שלך.
3. כדי למחוק את התכנית שבזכרון המחשב, הדפס _____ והקש _____ המחשב ידפיס _____.
4. כדי למחוק קטע מן התכנית שבזכרון המחשב, הדפס _____ והקש _____.
5. בעת הכנסת תכנית הדפסת PRINY בעוד שהתכוונת להדפיס PRINT. כדי לתקן טעות זו עליך להדפיס _____ או _____ ואח"כ להדפיס T.
6. השתמש בגישה ישירה לאיזון פנקס ההמחאות שלך, המכיל 3 מספרים:
 יתרה: 172.16
 המחאות: 114.14, 10.00, 3.52, 19.00, 13.50
 הפקדות: 87.57
7. חבר תכנית שתחשב את גובהם הממוצע של 10 ילדים המוזמנים למסיבת יום ההולדת של דני, גובהם של הילדים (באינצ'ים) לפי הסדר:
 45, 44, 41, 52, 49, 48, 46, 50, 40
 1.50 1.48 1.40 1.55 1.60 1.50 1.48 1.45 1.35
 כתוב כאן את תכניתך, לפני שתתחיל להריץ.

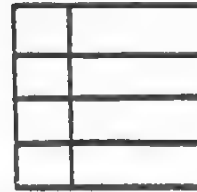
התשובות או הפתרונות מופיעים בסוף הספר (אם המורה שלך לא גזר אותם משם...).



תאים קטנים



תאר לעצמך שלמטה, בתוך המחשב, יש קבוצה של קופסאות קטנות, המחולקות לשני תאים, כשהתא השמאלי של כל קופסא יכול להכיל סימן המכונה משתנה והתא הימני יכול להכיל מספר, המכונה ערך המשתנה.



המשתנה נכנס כאן
יש המכנים זאת
שם המחר

הערך נכנס כאן
יש המכנים זאת
הנתון לשם הנתון

A	32
B	6.5
X	3
P	-10



יש לנו מספר סימני משתנים שהוכנסו לקופסאות, וערכים שהוצבו לאותם משתנים. למשתנה A הצבנו את הערך 32. פירוש הדבר הוא פשוט ש-A=32. למשתנה B הצבנו את הערך 6.5, כלומר B=6.5. בדומה לכך, X=3, P=-10.



זו הדרך בה ניתן להביא את המחשב לידי כך שימלא את התאים הקטנים, כלומר להציב ערכים למשתנים.

LET

וודאי כבר שמת לב
כולים להיות
אותיות האלפבית.

NEW

```
OK
10 LET A=32
20 LET B=6.5
30 LET X=3
40 LET P=-10
50 PRINT A, B, X, P
RUN
32          6.5          3          -10
```

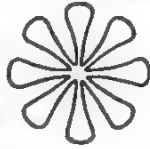
לאחר הרצת (RUN) התכנית התאים ייראו כדלהלן:

A	32
B	6.5
X	3
P	-10

OK

50 PRINT A, B, X, P

בהוראה זו אין המחשב מתבקש להדפיס את האותיות A, B, X, P, מפני שאינן מחרוזות בתוך מרכאות. בהוראה זו מבקשים מהמחשב להדפיס את הערכים המוצבעים למשתנים אלה.





מה?? שני סימני A, שלכל אחד מהם הוצב ערך אחר על-ידי שורות 10 ו-30? אם תסתכל בשימת לב תראה שהערך שהוצב על-ידי שורה 30 הוא הערך (15) שבו משתמשת שורה 40 כדי לערוך את החישוב האריתמטי ולהדפיס את התוצאה. ה-BASIC משתמשת תמיד בערך האחרון המוצב למשתנה. למעשה, כל ערך חדש המוצב למשתנה, ממלא את מקומו של הערך הקודם לאותו משתנה. הערך הקודם נעלם לעולם ועד, אלא במקרה שנערכת הצבתו מחדש.

להלן שיחזור (trace) התכנית, שאותה מבצע המחשב לפי סדר המספרים. המחשב מעדכן את ערכי המשתנים לאחר ביצוע כל הוראה והוראה. אם עדיין לא שמת לבך לכך, הרי המחשב הוא מהיר מאוד ומתחיל להדפיס את תוצאות התכנית שלנו כמעט בעת ובעונה אחת עם הקשתך על קלידי ה-RUN וה-RETURN.

תוכנית-RETURN

באור

הוראות

משתנים וערכים

ערכו של A, לאחר שהמחשב מבצע את ההוראה שקיבל בשורה 10.

ערכו של A עדיין נותר 5, לאחר שהמחשב מבצע את האמור בשורה 20 ומציב למשתנה B את הערך 10.

ערכו הקודם של A הוחלף בערך אחר, 15, לאחר שהמחשב מבצע את ההוראה שבשורה 30. ערכו של B לא השתנה.

ערכי המשתנים נשארים כמות שהם, כשהמחשב מבצע את שורה 40. הערכים של A ו-B משמשים למחשב לחישוב A-B ולהדפסת התוצאה.

10 LET A=5

A	5
---	---

20 LET B=10

A	5
B	10

30 LET A=15

A	15
B	10

40 PRINT A-B

A	15
B	10

RUN

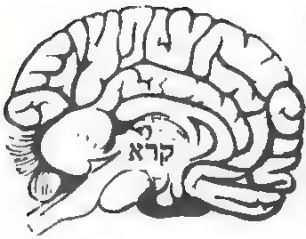
5

OK

זהו השיחזור

זהו השיחזור

חזרה על האנשה הנעירה



הנה עיצה טובה למתכנת הנבון. לאחר שהרצת תכנית והמחשב סיים והשיב OK, נשארים ערכיהם של כל המשתנים בתכנית בתוך הקופסאות. פירוש הדבר, שאתה יכול למצוא מה היו הערכים האחרונים ששימשו את המחשב בעת שסיים להריץ את התכנית (אילו ערכים נותרו בקופסאות בסופו של ה-RUN). תוכל לגלות זאת באמצעות השימוש בגישה ישירה הואיל והגישה הישירה אינה משתמשת במספרים סידוריים. אין לכך השפעה על התכנית שהכנסת זה עתה לזיכרוננו של המחשב.

השתמש בגישה ישירה כדי לגלות מהם הערכים שעדיין אצורים בזיכרוננו של המחשב. (אל תדפיס !NEW)

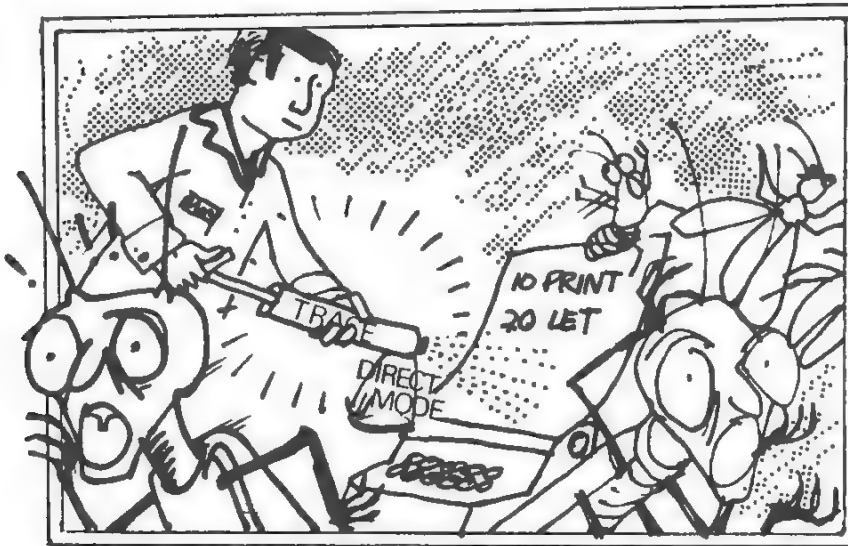
בצע

PRINT A,B

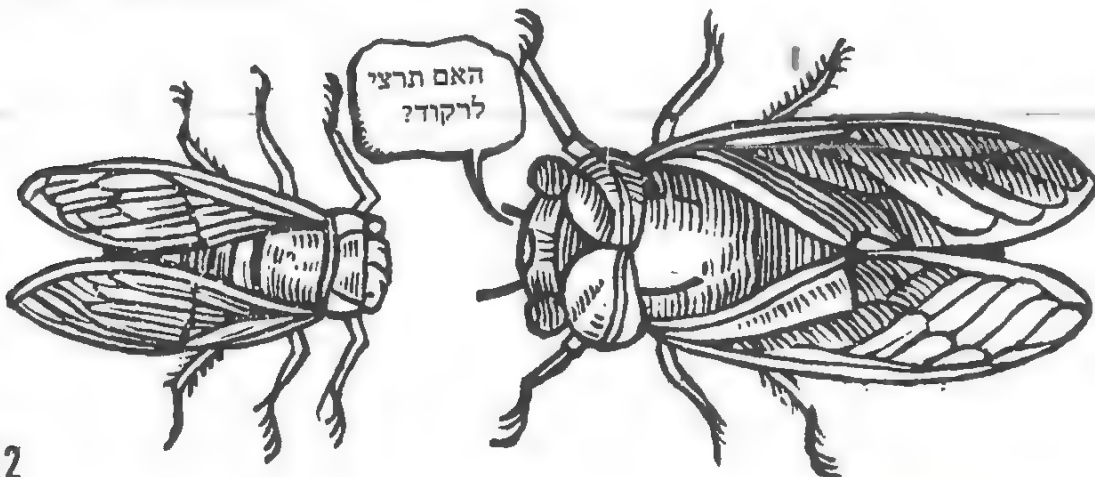
15

10

OK



בהשתמשך. בטכניקת השיחזור ובגישה ישירה אתה עשוי להיתקל גם בטעות כלשהי בתכניתך. טעות העשויה להיות טעות בהדפסה, טעות בשימוש ב-BASIC או אף טעות בתפיסה כיצד יש להביא את המחשב לעשות כבקשתך.

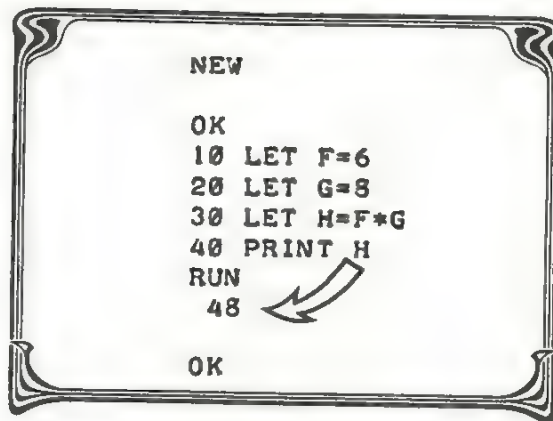




כפי שהינך למד מן הדוגמא האחרונה, לכל משתנה יש רק ערך אחד בכל פעם, והערך האחרון שהוצב לו, הוא שיירשם בקופסא לצד אותו משתנה. הערך האחרון ממלא את מקומו של הערך שלפני האחרון שהוצב, ועל כן כל ערך קודם שהוצב לאותו משתנה מבוטל לעולם.



בפרק זה יש מונחים ורעיונות חדשים לרוב. קרא ובצע יותר מפעם אחת!



הבט בשיחזור זה כדי לראות מה קורה בדיוק כשהתכנית מורצת.

באור

משתנים וערכים

הוראות

10 LET F=6

F	6
---	---

הערך שהוצב ל-F

20 LET G=8

F	6
G	8

הערך שהוצב ל-G

30 LET H=F*G

F	6
G	8
H	48

הערך שהוצב ל-H, תוך כדי שימוש בערכים של F ו-G לצורך החישוב האריתמטי.

40 PRINT H

F	6
G	8
H	48

המחשב מדפיס את הערך של H. שים לב לעובדה שכל הערכים מצויים בקופסאותיהם.

RUN
48

OK

זהו השיחזור



תרגול נוסף? בסדר.

NEW

OK

10 LET A=2

20 LET B=3

30 LET C=4

40 LET D=5

50 PRINT A+B+C+D, A*B*C*D, A*(B+C), (A+B)/(C+D)

RUN

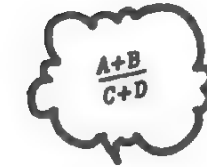
14

120

14

.555556

OK



LET

ועתה, ערוך שיחזור לתכנית זו.

באור

משתנים + ערכים

הוראות

10 LET A=2

A	
---	--

20 LET B=3

A	
B	

30 LET C=4

A	
B	
C	

40 LET D=5

A	
B	
C	
D	

50 PRINT A+B+C+D, ETC.

A	
B	
C	
D	

אל תשכח לתרגל!
עשה זאת מדי יום
בדומה!

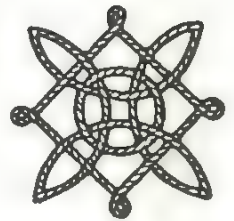


האם זכרת? ערכי המשתנים אינם מתחלפים כשמבוצעת הוראת PRINT, אפילו אם הערכים משמשים לחישובים (לפתרון בעיות אריתמטיות).

מחרוזות (STRINGS)



משמעותן של מחרוזות עמוקה יותר מאשר הצבת מספרים בתוך תאים. ערכו של משתנה עשוי להתבטא באמצעות מחרוזות ולא רק על ידי מספר. כדי שה-BASIC ידע שאנו עוסקים במשתנה מחרוזת, יסתיים שם המשתנה ב-S. לדוגמא, AS הינו הוראת LET. המחרוזות המוצבת למשתנה המחרוזת כלואה בתוך מרכאות, בדיוק כמו הוראת PRINT-ה.



NEW

```
OK
10 LET CS="VERY"
20 LET ES="GOOD"
30 PRINT CS; CS; ES
RUN
VERYVERYGOOD
```

CS	VERY
ES	GOOD

OK

טוב, נוכל לנסות זאת גם בדרך אחרת. נחליף את שורה 30 בשורה 30 חדשה ונחליף את סימני הנקודה פסיק בפסיקים וכל זאת מבלי להדפיס NEW.

```
30 PRINT CS, CS, ES
RUN
VERY          VERY          GOOD
```

OK

הממ.... עדיין לא משביע רצון. החלף שורה 10 ושורה 30 כדלהלן:

```
10 LET CS="VERY "
30 PRINT CS; CS; ES
LIST
      ↑      ↑
      ושוב נקודה פסיק

10 LET CS="VERY "
20 LET ES="GOOD"
30 PRINT CS; CS; ES
OK
RUN
VERY VERY GOOD
```

OK

נקווה שהתרגיל הקנה לך מושג כיצד מציבים מחרוזות למחרוזת משתנה וכן את הטכניקה להצבת מחרוזות זו לצד זו בהוראת PRINT-ה. כמובן שעוד נותר לומר דברים רבים בנושא המחרוזות. סמוך עלינו שעוד נרחיב את הדיבור בנושא!



נראה אם הבנתי את העניין: C שיהיה משתנה מספרי בעוד CS יהיה משתנה מחרוזת.





הלט (INPUT)

שיטה נוספת להכנסת סימני המשתנים לתא והצבת ערכים מתאימים היא עלידי השימוש בהוראת ה-INPUT (קלט). בתכנית הקטנה הבאה נסה להשתמש בהוראת ה-INPUT.

NEW

```
OK
10 INPUT CS
20 INPUT ES
30 PRINT CS, CS, ES
RUN
?
```

המחשב יושב ומחכה שתתן הוראת INPUT, כלומר מחרוזת שתוצב ל-CS.

הוראת INPUT זו גורמת למחשב להדפיס ? בפעם הראשונה בעת הרצת תכניתך.

הוראת INPUT זו גורמת להופעתו של ה-? השני.

האם הבחנת בהופעת סימן השאלה? הוראת ה-INPUT (קלט) גורמת למחשב להדפיס את סימן השאלה. המחשב מצפה לקלט, כלומר להכנסת דבר מה לתוך הקופסא המסומנת ב-CS. אתה צריך להגיב בהדפסת VERY (או כל מחרוזת אחרת) ולאחר מכן ללחוץ על RETURN.

(זהו המשכה של אותה הרצה).

```
RUN
? VERY
?
```

הדפסנו VERY ולחצנו על RETURN.

CS	VERY
----	------

המחשב ביצע CS = VERY ואז גורמת הוראת ה-INPUT שבשורה 20 להדפסת ה-2 השני והמחשב שוב מצפה להוראת INPUT כדי להציב ערך ל-ES.

עוד סימן שאלה? כן, שורה 20 בתכנית מורה למחשב לבקש מחרוזת עבור ES, ועליך להפעיל כאן דימיוןך. הדפס GOOD (או הייה מקורי) ולחץ על - RETURN.

```
RUN
? VERY
? GOOD
VERY
```

(ושוב אותה הרצה, חברה).

הדפסנו GOOD ולחצנו על RETURN.

ES	GOOD
----	------

OK

המחשב מבצע שורה 30 ומדפיס זאת.

המחשב האמין והזקן מדפיס את המחרוזת המכונה CS פעמיים, ואת המחרוזת התואמת את ES פעם אחת, כפי שמורה הוראת ה-PRINT שבשורה 30.

אם נרצה לקבל רווח לפני או אחרי המחרוזת שאנו מכניסים לאחר סימן שאלה של INPUT, עלינו להכניס את המחרוזת בתוך מרכאות. היינו יכולים להכניס את המלה VERY בתוך מרכאות ולכלול גם רווח בתוך המחרוזת. ראשית, שנה שורה 30, כך שמשתני המחרוזת מופרדים עלידי פסיקים ולא נקודות פסיק, ואז חזור והרץ את התכנית.

```
30 PRINT CS; CS; ES
RUN
? "VERY "
? GOOD
VERY VERY GOOD
```

OK

היי, אני הוא סימן השאלה INPUT המצפה לך!

קלט עם מחרוזות



אך סימן השאלה בפני עצמו אינו מספק מידע רב ביותר. אינך יודע מה עליך להשיב לסימן שאלה INPUT, אלא אם הינך יודע מהו נושא התכנית. בדרך זו אתה משיג רמז באשר לצרכיו של המחשב ל-INPUT.

NEW

OK

```
10 INPUT "WHAT IS YOUR NAME"; NS
20 PRINT NS; " IS YOUR NAME."
```

שים לב לנקודה פסיק.

שים לב לרווח

שים לב לנקודה פסיק

אמור למחשב מהו שמך (הדפס אותו, טיפשונו, ואל תשכח להקיש על RETURN) כשהוא שואל אותך זאת.

RUN

אתה משיב ← הוא שואל ←

```
WHAT IS YOUR NAME? JERALD R. BROWN
JERALD R. BROWN IS YOUR NAME.
```

OK

NS	JERALD R. BROWN
----	-----------------

מחרוזת ה-INPUT שלך הוצבה ל-NS.



שים לב לכך שעליך להכניס את מחרוזת ה-INPUT

"WHAT IS YOUR NAME"

למראות, ושכין מחרוזת ה-INPUT ומשתנה ה-INPUT עליך להשתמש בנקודה פסיק (;).

```
10 INPUT "WHAT IS YOUR NAME"; NS
```



מחרוזת ה-INPUT



משתנה ה-INPUT

המשך יבוא....



ניסינו להשתמש בפסיק ובהמשך נראה מה קרה: (אל תעשה זאת.
לשם שינוי די אם תסתכל במה שעשינו אנחנו)

```
10 INPUT "WHAT IS YOUR NAME", N$
RUN
```

```
?SN ERROR IN 10
OK
```

ובכן. ה-BASIC לא התלהב מכך כלל
וכלל. זוהי אותה הודעה על שגיאת
תחביר.

אם תבדוק את שורה 20 בתכניתנו, תווכח שאנו משתמשים בנקודה
פסיק בהוראת ה-PRINT, כדי שכל חלקיה יופיעו צפופים.

```
20 PRINT N$; " IS YOUR NAME."
```

(נסה לכתוב פסיק במקום נקודה פסיק, וראה את ההבדל.)

אינני מחרוזת,
כך שאל תשחיל עלי!



ועתה, מבלי להדפיס NEW, הוסף לתכנית את השורות
30 ו-40. הרץ עכשיו את התכנית ונהל דרשיח קצר
עם המחשב שלך.



```
30 INPUT "HOW OLD ARE YOU"; A
40 PRINT N$; ", YOU ARE"; A; " YEARS OLD."
```

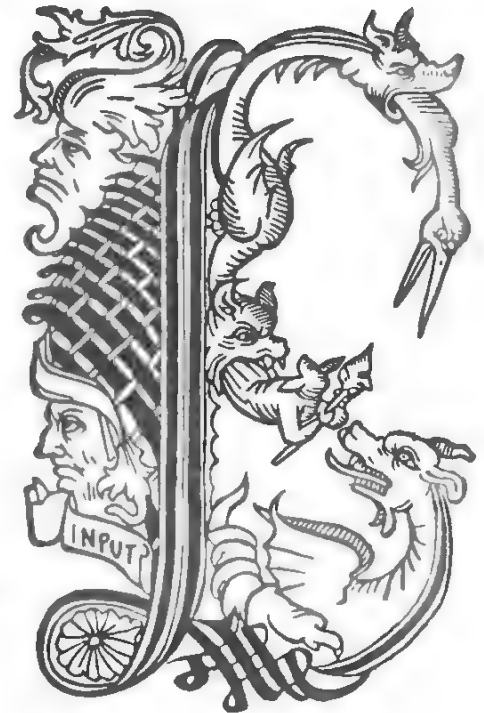
השתמשנו במשתנה מספרי, הואיל
והתכנית דורשת מספר. אך יכולנו
להשתמש גם במשתנה מחרוזת
הואיל והמספר לא יוכל לשמש בכל
סוג של חישוב.

תן דעתך לפסיק ולרווח שבתוך המרכאות.

```
RUN
WHAT IS YOUR NAME? JACK
JACK IS YOUR NAME.
HOW OLD ARE YOU? 24
JACK, YOU ARE 24 YEARS OLD.
```

מישהו הדפיס את שמו וגילו
אחרי סימני השאלה.

OK





ועתה, אם הינך מעוניין לראות את התכנית בשלמותה, הדפס LIST.

LIST

```
10 INPUT "WHAT IS YOUR NAME"; NS
20 PRINT NS; " IS YOUR NAME."
30 INPUT "HOW OLD ARE YOU"; A
40 PRINT NS; ", YOU ARE"; A; "YEARS OLD."
OK
```



והרי דרך נוספת להכנסת תכנית. זוהי דרך מיושנת מאוד.

NEW

```
OK
10 PRINT "WHAT IS YOUR NAME";
20 INPUT NS
30 PRINT NS; " IS YOUR NAME."
40 PRINT "HOW OLD ARE YOU";
50 INPUT A
60 PRINT NS; ", YOU ARE"; A; "YEARS OLD."
RUN
WHAT IS YOUR NAME? BILL
BILL IS YOUR NAME.
HOW OLD ARE YOU? 22
BILL, YOU ARE 22 YEARS OLD.
```

NS	BILL
A	22

OK

הוראת ה-LET
מיועדת להצבת ערכים מפורשים למשתנה מסוים, והינה אופציונאלית.
ביטוי אלגבראי = (משתנה) LET (מס' סידורי)

```
10 LET X = 3
20 LET Z = X*5
30 NS = "NAME"
```

רק BASIC PLUS

צד = כל המשתנים, LET הכפל - 33 X, Y, Z LET

הוראת ה-INPUT

מבקשת נתונים מן המסוף (?)

שם או שמות המשתנים או (LIST) INPUT

שם או שמות המשתנים ("מחרוזת") INPUT

```
10 INPUT X
20 INPUT X,Y,Z
30 INPUT "WHAT IS YOUR NAME"; NS
```



קרא

הוראת ~ DATA - READ

תוכל לשכנע את המחשב למלא את אותן הקופסאות הקטנות, דהיינו להציב ערך למשתנים השונים, בדרך שלישית. ראשית, השתמשנו בהוראת ה-LET. לאחר מכן גדרשנו ל-INPUT. ועתה אנו משתמשים ב-READ DATA. שיטה זו מיישמת את צירופן של שתי הוראות BASIC, המופיעות תמיד זו לצד זו. הוראת ה-READ והוראת ה-DATA הן ידידות טובות ולמעשה הן יכולות לשמש מעין חבורה, כש-READ הוא המנהיג של DATA רבים.

תרגל כל אחת מן הגירסאות של התכנית המובאות בהמשך וגלה כיצד פועלות שתי ההוראות בצוותא כדי להציב ערכים למחרוזות או למשתנים. כדי לפשט בתחילה את הדברים, נשתמש רק בערך DATA אחת ויחיד, משתנה או מחרוזת, בהוראת ה-DATA.

משתנים מספריים



```
10 READ N
20 PRINT N
30 DATA 58
RUN
```

OK

```
10 DATA 58
20 READ N
30 PRINT N
RUN
```

OK

```
10 READ N
20 DATA 58
30 PRINT N
RUN
```

OK

```
10 READ N
20 PRINT N
30 DATA FIFTY EIGHT
RUN
```

?SN ERROR IN 30
OK



משתנים מספריים

שים לב לכך שהמחשב מתייחס לשתי המלים FIFTY EIGHT כאל מחרוזת אחת ויחידה, כולל הרווח שבין המלים.

```
10 READ AS
20 PRINT AS
30 DATA FIFTY EIGHT
RUN
```

OK

```
10 DATA FIFTY EIGHT
20 READ AS
30 PRINT AS
RUN
```

OK

```
10 READ AS
20 DATA FIFTY EIGHT
30 PRINT AS
RUN
```

OK

```
10 READ AS
20 DATA 58
30 PRINT AS
RUN
```

OK

הרשו נא לי להפנות את תשומת לבכם לעובדות: (1) מחרוזות יכולות להיות מספרים, ו-2 לדוגמא, ל-AS יכול להיות 58 כמחרוזת, אך אין להשתמש ב-AS=58 כבערן (בחישובים אריתמטיים, לדוגמא) ו-3) אני 36-24-36... האם תרצה שאחזור על כל זאת?



הם עובדים יחד ומציבים ערכים ומחרוזות למשתנים.



כפי שתבין מדוגמאות אלו, תציב הוראת ה-READ את הערך או המחרוזת בהוראת ה-DATA למשתנה שלו (משתנה ה-READ). לא משנה היכן מוצבת הוראת ה-DATA בתוך התכנית (בתחילה, בסוף או באמצע), מציב המחשב את האיבר הראשון בהוראת ה-DATA הראשונה ולמשתנה ה-READ הראשון.



קרא

בהתבוננך בדוגמא האחרונה, שים לב שהאיברים בהוראת ה-DATA מופרדים על-ידי פסיקים, אך אין פסיק לאחר האיבר האחרון.



10 READ X\$, Y\$, Z\$, K, L



ועתה נסה את התכנית הבאה, המוכיחה שהוראת READ בודדה עשוייה לשמש להצבת ערכים מספריים ומחרוזות גם יחד, למשתני ה-READ המתאימים באותה הוראת READ עצמה.

NEW

OK

```
10 READ BS, CS, A, B, C
20 PRINT A; BS; B; BS; C; CS; A+B+C
90 DATA ADDED TO, GIVES YOU, 5, 8, 10
RUN
5 ADDED TO 8 ADDED TO 10 GIVES YOU 23
```

OK

שני אלה הינם משתני מחרוזות.

10 READ BS, CS, A, B, C

ואלה הינם משתנים מספריים.



המחשב מציב את המחרוזות או את הערכים המספריים לפי התור למשתנים שבהוראת ה-READ. יחד עם זאת, המחשב הוא קפדן גדול. הוא יציב ערכים מספריים רק למשתנים מספריים (אלה המופיעים בלא \$) משתני מחרוזות (עם \$ בסופם) יקבלו מספרים כמחרוזות וגם כמלים ומספר סמלים, כדוגמת סימני פיסוק. אך המחשב לא ישלוף ויבחר. ה-DATA חייבת להופיע בסדר הזהה לזה של המשתנים: ערכים מספריים למשתנים מספריים ומחרוזות למשתני מחרוזות. תפסת? שים לב לכך שכל איבר בהוראת ה-DATA מופרד על ידי פסיק, אך אין פסיק אחרי האיבר האחרון.

החלף את שורה 10 בתכנית בשורה 10 המובאת להלן. מדוע לא תוכל להריץ אותה?

```
10 READ A, B, C, BS, CS
```

(אם תרצה לבדוק את התכנית כולה, הדפס LIST. אם אתה בעל ביטחון עצמי תוכל להסתפק בהדפסת RUN).

LIST

```
10 READ A, B, C, BS, CS
20 PRINT A; BS; B; BS; C; CS; A+B+C
90 DATA ADDED TO, GIVES YOU, 5, 8, 10
OK
RUN
```

?SN ERROR IN 90
OK

שים לב שהמחשב חושב שטעית בסדר ושהכנסת את הנתונים בהוראת ה-DATA.

900 DATA BEAT, STAG, DEER, 66, 89





כפי שכבר אמרנו, הוראת ה-READ לא תבחר באיברים שבהוראת ה-DATA. האיברים ב-DATA חייבים להופיע בסדר הזה לסדר המשתנים - רק מספרים עבור משתנים רגילים ומחרוזות למשתני מחרוזות (אותם משתנים שבהם מופיע \$ אחרי האות). ועוד דבר שעליכם לדעת בקשר להוראות DATA. גם אם משתמשים בכמה הוראות READ במסגרת תכנית אחת, הן עשויות ליטול את הערכים שלהן, לפי התור, מהוראת DATA אחת ויחידה. לאחר שמשתמשים בכל ה-DATA מתוך הוראת DATA אחת, דהיינו כשכל האיברים הוצבו פעם אחת ויחידה למשתנים, ממשיך המחשב אל הוראת ה-DATA הבאה. גם אם הם ממוקמים במרחק זה מזה בתוך התכנית, מחפש המחשב אחר הוראות ה-DATA לפי המספר הסידורי שלהן, ומתחיל בהוראת ה-DATA בעלת המספר הסידורי הנמוך ביותר.

איך ההדפסה שלך?



NEW

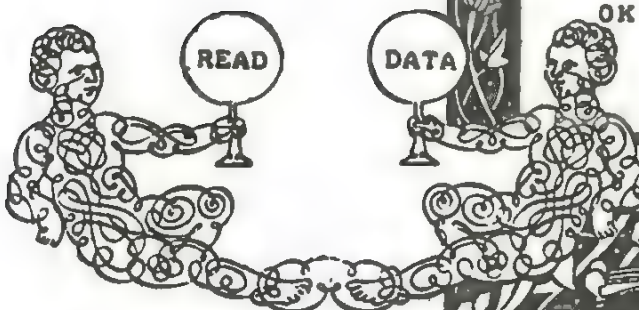
OK

```
10 READ A
20 PRINT A
30 READ B
40 PRINT B
50 READ C
60 PRINT C
70 READ D
80 PRINT D
90 READ A
100 PRINT A
900 DATA 832, 5009, 32
910 DATA 581, 200
```

RUN

832	←	900 DATA-N, A ערך
5009	←	900 DATA-N, B ערך
32	←	900 DATA-N, C ערך
581	←	910 DATA-N, D ערך
200	←	910 DATA-N, A ערך חדש של A

OK



השימוש במרחאות

בהוראות ה-DATA



קרא

הוראות READ ו-DATA פועלות באותו אופן גם על מחרוזות. רק הערה קטנה - אם אתה רוצה במחרוזות שבתוכה פסיקים, עליך להשתמש במרכאות, בדיוק כפי שאתה עושה בהוראות LET או במחרוזות INPUT.

איבר ראשון

איבר שני

30 DATA "PARSLEY, SAGE, ROSEMARY, AND WINE", ARE FINE
40 DATA TO DINE

ערך DATA שלישי



NEW

OK

10 READ X\$, Y\$, Z\$
20 PRINT X\$, Y\$, Z\$
30 DATA "PARSLEY, SAGE, ROSEMARY, AND WINE", ARE FINE
40 DATA TO DINE

RUN

PARSLEY, SAGE, ROSEMARY, AND WINE

ARE FINE

TO DINE

OK



הוראת ה-READ קוראת נתונים והופכת אותם למשתנים מסוימים או למשתני מחרוזת מהוראות ה-DATA. מתחילה בהוראת ה-DATA הראשונה בתכנית וממשיכה לעבור על כל הוראות ה-DATA.

10 READ X, Y, Z
20 READ N\$, A\$, Z

הוראת ה-DATA מספקת נתונים שיקראו באמצעות הוראת ה-READ. מספר סידורי DATA מספריים או מחרוזות, המופרדים על ידי פסיקים.

10 DATA 35, 16, -72, BINGO
20 DATA JEROME, MOLLY, ALBERT

במחרוזות - רצוי להשתמש בסימני (:) או (,) בתוך המרכאות.

30 DATA "INSTANT BASIC, BY BROWN, FINKEL"

מחרוזות עם רווחים



מקרה אחר המחייב שימוש במרכאות בהוראת DATA, הוא כשתרצה רווח אחד או יותר לפני או אחרי התווים שבמחרוזות. כפי שראינו קודם לכן, רווחים בין מלים או לבין תווים כלולים אף הם במחרוזות. רק במקרה שהינך רוצה ברווחים, לפני או אחרי המחרוזות, יהיה עליך להשתמש במרכאות בתוך הוראת DATA. בדוק זאת.



הדפס תכנית זו ואז הרץ אותה ואת מספר השינויים שהוכנסו בה.

NEW

OK

10 READ AS, BS, CS

20 PRINT CS; BS; AS

30 DATA GREAT, THIS, AIN'T

RUN

AIN'TTHISGREAT

← לא כל כך, למען האמת...



...שלוש מחרוזות בהוראת DATA

OK

X

30 DATA GREAT, THIS, AIN'T

RUN

AIN'TTHISGREAT

← החלף את שורה 30 בהוראת DATA בעלת רווחים לפני או אחרי המחרוזות. הרווחים לא נכללו במחרוזות שהוצבו למשתני המחרוזות, אך אל יאוש.

OK

30 DATA GREAT, " THIS ", AIN'T

RUN

AIN'T THIS GREAT

← החלף שוב את שורה 30 בערך DATA, המשתמש במרכאות סביב ערך המחרוזת, כדי להורות למחשב לכלול רווחים לפני ואחרי המלה, כחלק מן המחרוזות.

OK

LIST

← זה פועל!

10 READ AS, BS, CS

20 PRINT CS; BS; AS

30 DATA GREAT, " THIS ", AIN'T

OK

← המחשב יודע שאנו רוצים לכלול הכול בתוך המרכאות שבתוך המחרוזות, כלומר גם את הרווחים.

1. מנה את שלושת סוגי ההוראות המשמשות להצבת ערכים למשתנים (מספריים או מחרוזות)
2. כשהינך משתמש בהוראת ה-LET, חייב מידע משתנה המחרוזת להכליל ב-
 3. לאחר הרצת תכנית, המשתנים מכילים עדיין את הערך האחרון. כיצד אתה מוצא ערכים אלה?
4. מלא את תאי השיחזור. A B C D

```

10 LET A = 12
20 LET B = 14
30 READ C
40 PRINT A + C
50 LET D = B + C/6
60 PRINT D
70 DATA 16, 13, 14

```

5. כתוב תכנית אשר תקרא ותדפיס את המידע האצור בהוראת DATA
 זר: 23, 800-555-1212, DATA SHERRY DELIGHT.
6. כתוב תכנית המזרזת את המשתמש להכניס את השם, תאריך הלידה והמזל שלו. לבסוף, הדפס את התוצאה המופיע להלן:
 SHERRY DELIGHT, YOUR SIGN IS LEO SINCE YOU WERE BORN 8/15/60.
7. מתקן חדש לחיסכון מים בשרותים בביתך אמור לחסוך 150 גאלוני מים בחודש. מחירו \$4.80. מחיר המים הוא \$0.48 ל-100 מטרים מעוקבים (א) כמה חודשי שימוש או אי שימוש במים יצדיקו את ההשקעה במחיר המתקן?
 (ב) יחד עם המתקן מקבלים כתב אחריות ל-12 חודשים בלבד. האם הרכישה משתלמת במונחים כלכליים טהורים?
 (אתה יכול וצריך ליישם אותה שיטת פתרון ביחס למתקנים חוסכי אנרגייה אחרים, המוצעים למכירה בשוק.)
8. בעידן זה של מחסור באנרגייה ובמים, אנו בחוף המערבי מפתחים מודעות גדולה והיינו רוצים להתחלק עמך ברעיונות שלנו (הנה מגיע תורך!) מבחינת מים. אנו יודעים מהן העובדות והמספרים המתייחסים לצריכת המים של משפחות רבות וכדומה. בתחתית העמוד מופיעה טבלה של צריכה ממוצעת בפעילויות השונות. המספרים המדויקים יהיו כנראה שונים במעט ולכן עליך לשנות את הטבלה כך שתתאים לצרכיך. העזר בטבלה ובמחשב הנוח שלך לכתובת תכנית שתחשב את צריכת המים החודשית של משפחתך. השתמש בהוראות INPUT לשם הכנסת המשתנים ובהוראות DATA להכלת הקבועות (מן הטבלה). תכנית זו עשויה להיות ארוכה, אך למעשה אינה קשה כדי כך.



מקלחת	6 גאלונים/דקה	אמבטיה	20 גאלונים
הדחת כלים	15 גאלונים	מדיח כלים	16 גאלונים
משטף	6 גאלונים	(בדוק הצריכה בביתך)	
השקייה	בבית ממוצע	רחיצת מכונת	35 גאלונים
	10 גאלונים/דקה		
שונות	להערכתך/ליום		

ברגע שהתכנית פועלת, תוכל לשנות את הנתונים (לצמצם צריכת המים במשטף מקלחות וכו'), ולהוכיח כמה מים אפשר לחסוך. (יש משפחות בקליפורניה המוגבלים לצריכה של 40-45 גאלוני מים לאיש, ליום האם תוכל להתגבר על כך?)



לולאות לולאות



או בתל"ס אחרות GOTO ...



בצע

NEW

```

OK
10 PRINT "THIS IS A LOOP."
20 GOTO 10

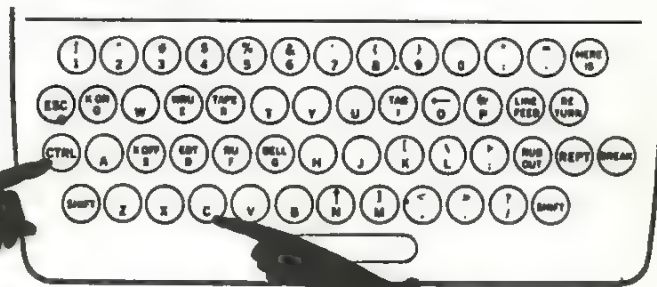
```

GO TO 10 מורה למחשב להתקדם אל ההוראה שמספרה הסיידורי 10, ולהמשיך להריץ את התכנית בהתאם למספרים הסיידורים.

חכה! עמוד! עצור! הפסק! חדל!

לפני שהינך מריץ את התכנית חפש את הקליד CONTROL (הפיקוח) (CTRL) והקליד C.

מצאת? בסדר, ועכשיו הרץ את התכנית. כשימאס לך להסתכל בתוצאות, לחץ על הקלידים CONTROL ו-C בעת ובעונה אחת, כדי לשים קץ להרצת התכנית.



לחץ על CTRL ו-C בעת ובעונה אחת.

```

RUN
THIS IS A LOOP.
THIS IS A LOOP.
THIS IS A LOOP.
THIS IS A LOOP.
THIS IS A LOOP.
THIS IS A LOOP.
THIS IS A LOOP.

```

```

BREAK IN 10
OK

```

לחצנו על CTRL ו-C ביחד

המחשב אומר לך באיזו שורה נכנסו ה-CONTROL/C לתמונה. אילו המשכנו להריץ, המחשב היה מבצע את שורה 10.

אל פחד!
לחץ על CTRL/C

קרא

המחשב ביצע שוב ושוב את שורה 10 מפני שהוראת ה-GO TO הורתה למחשב לחזור לשורה 10, בכל פעם שהוא סיים את השורה 10 ועבר זוהי לולאה אינסופית הממשיכה ליצור מעגלים בחזרה על עצמה, עד שאדם נבון לוחץ על CTRL ו-C בעת ובעונה אחת. השתמש ב-CONTROL/C כל אימת שתמצא להפסיק או לעצור את המחשב בעת הרצת תכנית.

לולאה ממוספרת בעזרת GOTO הוראת



ועתה הכנס את תכנית ה"ספירה". זוהי עוד תכנית לולאה אינסופית
התכונן איפוא ללחץ על CTRL/C לאחר תחילת ההרצה.

NEW

```
OK
10 LET T=1
20 PRINT T,
30 LET T=T+1
40 GOTO 20
```

RUN

```
1
6
11
BREAK IN 20
OK
```

2
7
12

שים לב לפסיק בסופה של שורה 20, ולצורת
פיזור הוראת ה-RUN על פני העמוד. חבר 2
ועוד 2. מה עושה לתוצאה הפסיק שבסוף
השורה?

3
8

4
9

5
10

(האם זכרת ש-CONTROL/C עוצר את התכנית?)
החלף את שורה 20 ע"י הדפסת שורה 20 חדשה, שבסופה נקודה פסיק. (אל
תדפיס NEW פן תבטל את כל שאר התכנית!)

20 PRINT T;

הקש על LIST (אם הינך רוצה לראות את התכנית בשלמותה, הכוללת את
שורה 20 החדשה) או הרץ (RUN) אותה.

RUN

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30 31 32
BREAK IN 20
OK
```

רוצה לראות מה קורה כשאין פסיק או נקודה פסיק בסוף הוראת
ה-PRINT? אם כן, החלף שורה 20 כדלהלן:

20 PRINT T

RUN

```
1
2
3
4
5
6
7
8
```

BREAK IN 20
OK



1
2
3
4
5
6
7
8
9
10
11

הבה נעקוב אחר המחשב, בעת שהוא מריץ את התכנית האחרונה. עבור
על התכנית לפי סימון החצים.

התחל כאן

10 LET T=1

20 PRINT T

30 LET T=T+1

40 GOTO 20

GO TO ...

זוהי לולאה. זו לולאה "לעולם ועד".
היא ממשיכה וממשיכה וממשיכה וממשיכה ...
בעת שהתכנית מורצת, עד שתעצור אותה בעזרת -
CONTROL/C.

והנה גישה נוספת לעניין, השיחזור של התכנית. זכור שהוא משחזר את
הצעדים שהמחשב נקט לאורך התכנית שעיקר, ומראה לך אילו ערכים הוצבו
למשתנים בכל שלב ושלב. בעמודה המסומנת ב-T אנו מראים את ערכו של T,
לאחר שההוראה המופיעה באותה שורה בוצעה על-ידי המחשב. במלים אחרות
אנו מראים איזה ערך נמצא בתא המכונה T, לאחר שבוצעה כל הוראה. אתה
יכול לדמיין שאותו תא קטן ישמש תמיד את משתנה T - כך שרק הערך המוצב
ל-T משתנה מפעם לפעם.

הוראה

משתנים וערכים

באור

10 LET T=1

20 PRINT T

30 LET T=T+

40 GOTO 20

20 PRINT T

30 LET T=T+1

40 GOTO 20

20 PRINT T

30 LET T=T+1

40 GOTO 20

T	1
---	---

T	1
---	---

T	2
---	---

T	2
---	---

T	2
---	---

T	3
---	---

T	3
---	---

T	3
---	---

T	4
---	---

T	4
---	---

הצב ל-T את הערך 1.

הדפס את ערכו של T.

הגדל את T ב-1 (הוסף 1
לערכו הקודם של T).

חזור לתחילת הלולאה.

הדפס את ערכו של T.

הגדל את T ב-1.

חזור לתחילת הלולאה.

הדפס את ערכו של T.

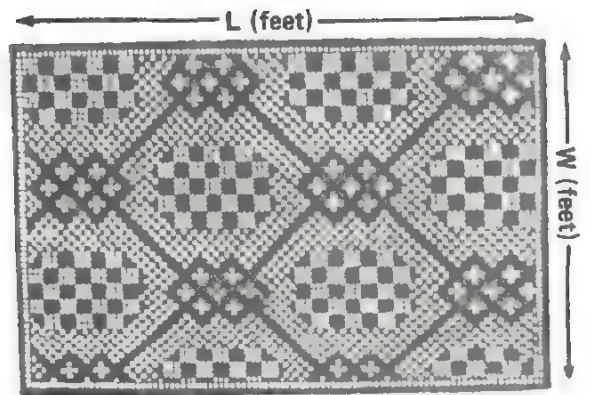
הגדל את T ב-1.

חזור לתחילת הלולאה.

וכר', וכר'...וכר'...

זהו השיחזור





NEW

```

OK
10 INPUT "LENGTH OF FLOOR IN FEET"; L
20 INPUT "WIDTH IN FEET"; W
30 PRINT "YOU NEED"; (L*W)/9; "SQ. YARDS OF CARPET."
40 PRINT
50 GOTO 10
RUN
LENGTH OF FLOOR IN FEET? 12
WIDTH IN FEET? 10
YOU NEED 13.3333 SQ. YARDS OF CARPET.

```

```

LENGTH OF FLOOR IN FEET? 16
WIDTH IN FEET? 18
YOU NEED 32 SQ. YARDS OF CARPET.

```

שורה רוח לזכותה של שורה 40

```

LENGTH OF FLOOR IN FEET? 16
WIDTH IN FEET? 24
YOU NEED 42.6667 SQ. YARDS OF CARPET.

```

```

LENGTH OF FLOOR IN FEET?

```

לחץ על RETURN כדי להחליף ממצב RUN שנעצר ומהוראת INPUT למשתמשים ב-BASIC PLUS - השתמשו ב-CONTROL/C לחיצה על RETURN מציבה אפס למשתנה מספרי ר' מחרוזת אפס" (מחרוזת נטולת תווים) למשתנה מחרוזת. זוהי האזהרה היחידה שתשמע.

האם תוכל לשער מה תפקידה של שורה 140?

אם ניחשת "לא כלום" כמעט וקלעת למטרה. היא מותירה שורה ריקה בהדפסה ובכך מפרידה בין כל הלולאות שבתכנית, וזאת כדי למנוע מכס בילבול יתר בעד הרצת התכנית או השימוש בה.

★ מיוחד למומחים: הוסף שורות לתכנית או הכנס בה שינויים מתאימים כדי שתוכל לחשב את מחיר השטיח אם מחירו של יארד מרובע הינו \$8.99, או להגיע לטבלה שתבטא את מחיריהם השונים של שטיחים.



הצעה: אל תשתמשו ב-CONTROL/C. הניחו לתכנית להתגלגל מספיק פעמים כדי להגיע לגבול העליון של הטיפול במספרים. אתה עלול לקבל הודעת Overflow error שתעצור את ביצוע התכנית.



NEW

```

OK
10 LET N=2
20 LET N=N+2
30 PRINT N;
40 GOTO 20
RUN

```

```

4 16 256 65536 4.29497E+09 1.84468E+19
70V ERROR IN 20 ←
OK

```

הוראת ה-overflow error. המספר גדול מדי אפילו עבור המחשב!

מה כל הקישקוש הזה?



אתה רואה שאחרי שהמחשב עבר את המספר 65536 הוא החל להדפיס תוצאות כדוגמת $4.29497E+09$. הדבר קרוי סימון הנקודה העשרונית - וזוהי למעשה צורת קיצור לביטוי מספרים גדולים מאוד או שברים עשרוניים קטנים ביותר. בסימון הנקודה העשרונית שני חלקים.

מעריך
 $E+00$
 מנטיסה

ה-E מראה היכן מתחיל המעריך, העשוי להיות חיובי או שלילי. תמיד תמצא + או - אחרי האות E.

הנה מספר דוגמאות למספרים הכתובים בצורה הרגילה, ובצורה של

הנקודה הצפה.

1,000,000,000 או 1000000000

סימון רגיל

ביליון

מעריך
 $1E+09$
 מנטיסה

נקודה עשרונית:



31,708,000,000,000,000,000

3170800000000000000000

נפח כדור הארץ בבושלים

בסימון הרגיל אפשר להשתמש בפסיקים

או

מעריך
 $3.1708E+22$
 מנטיסה

סימון הנקודה העשרונית:

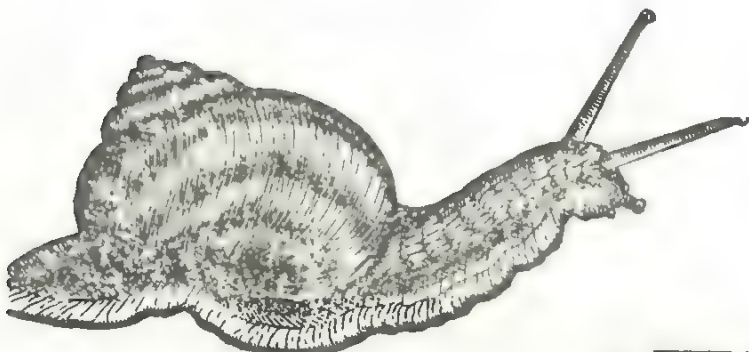
מהירות השבול במייל לשניה

.0000079

סימון רגיל

מעריך
 $7.9E-06$
 מנטיסה

סימון הנקודה העשרונית



הכללים:

צפה וקודה

ועתה אל הכללים: כיצד להפוך את סימון הנקודה העשורנית למספרים רגילים.

במקרה שהמעריך חיובי:

- (1) כתוב את המגטיסה בנפרד (כלומר, מבלי להתייחס למעריך ול-E).
- (2) הזז את הנקודה העשרונית של המגטיסה מספר מקומות ימין, בהתאם להוראת המעריך. באם יידרש, הוסף אפסים.

דוגמא:

1.0000000000

1000000000

הוספנו 9 אפסים.

ודוגמא נוספת:

3.1708E+22

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
3.170800000000000000000000.

הרשפנר 18 אפסימ.

במקרה שהמעריך שליילי:

- (1) כתוב את המגטיסה בנפרד (כלומר, מבלי להתייחס למעריך).
- (2) הוז את הנקודה העשרונית של המגטיסה מספר מקומות שמאלה, בהתאם להוראת המעריך. באם יידרש, הוסף אפסים.

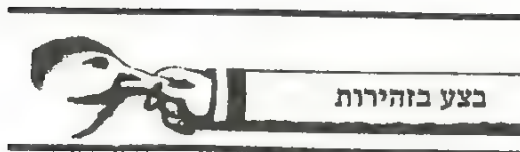
7.9E-06

6 5 4 3 2 1
.000007, 9

דגמא:

הוספנו 5 אפסים. 00000079.

ייתכן שכבר ניחשת שזוהי אותה נקודה עשרונית "צפה" העוברת ממקום למקום ובכך מצדיקה את שם הסימון. מעתה ואילך תדע היכן לחפש את השיטה בה הופכים את סימון הנקודה העשרונית לסימון רגיל.



NEW

OK

```
10 PRINT 1000000000000, 123456789, 3.987654321
20 PRINT .000009, .0000000654321, .700007007
30 PRINT 345.3456E4, .3E-9, 6.666666E-4
```

RUN

1E+12	1.23457E+08	3.98765
9E-06	6.54321E-08	.700007
3.45346E+06	3E-10	6.66667E-04

OK

LIST

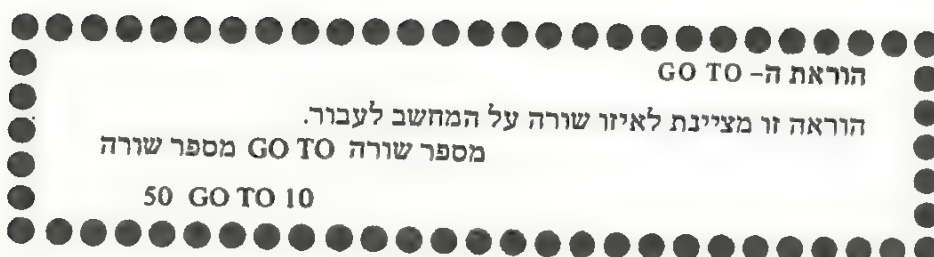
```
10 PRINT 1000000000000, 123456789, 3.987654321
20 PRINT .000009, .0000000654321, .700007007
30 PRINT 345.3456E4, .3E-9, 6.666666E-4
```

OK

כך שבעתיד תבין ללא כל קושי מה קורה כשהמחשב נותן לך תוצאה
בסימון הנקודה העשרונית.

האם כבר ערכת איזה ניסוי היום?

עשה זאת כאן! עשה זאת עכשיו!



פרק 3

1. כתוב את המספרים הבאים, בעלי הנקודה העשרונית הצפה, בצורה עשרונית רגילה.

8.967E5 _____
 1.00E15 _____
 1.0E-3 _____
 6.15762E10 _____
 3.87124E-6 _____

2. כתוב מספרים עשרוניים אלה בסימון הנקודה העשרונית.

1,786,432 _____
 3,134,575,321 _____
 .00012479 _____
 .42456 _____

אם התכנית שלך מחכה לנתונים כלולאת INPUT (?) והנתונים עדיין אינם בידך, כיצד אתה עוצר את התכנית?

4. אם תכניתך מורצת כלולאה אינסופית. כיצד אתה עוצר אותה? _____

5. כתוב תכנית שתדפיס חזקות.

תכנת כך שזה יימשך ללא הרף. הפסק כשהדבר יימאס עליך.

1	1
2	4
3	9
4	16
5	25
6	36

וכו'

6. כדי לסייע לך ולמשפחתך להגיע לשיטה המטרית ולטמפרטורות צלזיוס, כתוב תכנית שתערוך טבלה של טמפרטורות פארנהייט וצלזיוס, כשאתה מתחיל ב-30° פארנהייט. (נוסחה: $C = 5/9 (F - 32)$). הפסק את התכנית כשתרצה בכך.



עד כה השתמשנו במשתנים הפשוטים והבסיסיים ביותר: אות הלקוחה מן האלפבית, או למשתני מחרוזת, אות המלווה בסימן \$, המורה למחשב שעליו לצפות למחרוזת ולא לערך מספרי. יחד עם זאת, קיימים גם משתנים מסובכים יותר ולעתים יש סיבה טובה להשתמש בהם. בתכנית ארוכה ומסובכת אתה עשוי להזדקק ליותר מ-26 משתנים שונים. כדי להשיג בהירות בכל הנוגע לתכנית, תרצה בוודאי להדביק למשתנה תווית שתזהה אותו, או אולי אתה רוצה לתת לקבוצת משתנים דומים תווית זהות.

חוקים למשתני תווית.
הסימן הראשון שיבטא משתנה חייב להיות אחד מאותיות האלפבית. לאחר האות הראשונה, אתה רשאי להשתמש באותיות, מספרים או סימנים. (לדוגמא A3, A67, AL\$, S3\$ וכו'...)

ההסתייגות היא (ותמיד קיימת הסתייגות), שאסור לך להשתמש במלים ובקיצורים הנהוגים בשפת ה-BASIC לשם מתן הוראות והוראות הפעלה בתור משתנה או כחלק ממשתנה. מלים כדוגמת PRINT, INPUT, RUN, NEW או LET מהוות חלק משפת ה-BASIC ולכן אינן יכולות לשמש כמשתנים או כשמות נתונים. הרשימה השלמה של שמות אסורים, שאינם יכולים לשמש כשמות משתנים או כחלקם של משתנים, מופיעה במסגרת שבהמשך.

שמות משתנים חוקיים:

SALARY	S1
TAX	S2
PLAYER	T%
GUESS	

שמות משתנים אסורים:

LETTER
RUNNY NOSE
LISTERINE
SPRINT

להלן רשימת המלים השמורות בשפת BASIC. אסור להשתמש במלים או בשמות משתנים או בחלקי שמות משתנים

ABS	CLEAR	DATA	DIM	END	FOR	GOSUB
GOTO	IF	INPUT	INT	LET	LIST	NEW
PRINT	READ	REM	RESTORE	RETURN	RND	
RUN	SGN	SIN	SQR	STEP	STOP	TAB
THEN	TO	USR				
ASC	AND	ATN	CHR\$	CLOAD	CONT	COS
CSAVE	DEF	EXP	FN	FRE	INPUT	LEFT\$
LOG	MID\$	NULL	ON	OR	NOT	OUT
POKE	POS	RIGHT\$	SPC	STR\$	TAN	VAL
						WAIT



קרא

יש מילכוד בשימוש ביותר משני תווים בשמו של משתנה. שפת ה-BASIC בסגנון ALTAIR מתחשבת רק בשני התווים הראשונים בשמו של משתנה. הדבר עשוי להביא לתוצאות מעגיינות.



בצע

נסה תכנית זו תוך כדי שימוש במשתנים.

NEW

```
OK
10 LET JOHN=15
20 LET FRUMP=13
30 LET GERTRUDE=14
40 PRINT JOHN; FRUMP; GERTRUDE
50 PRINT JOE; FRANNY; GENE
RUN
```

JO	15
FR	13
GE	14

15 13 14 ← שורה 40 הדפיסה זאת

15 13 14 ← אך גם שורה 50 הדפיסה זאת

OK

”ועוד רעיון טוב, שלמרבה הצער אינו בא לידי ביטוי מעשי, הוא השימוש בשמות הנתונים. בכל שפות המחשב המתוחכמות אתה רשאי לקרוא לילד בשמו. אם הינך משתמש במשתנה כדי לשמור על סך הכול (TOTAL) אתה יכול לכנות זאת TOTAL. הדבר מסייע למתכנת לזכור מה מתרחש היכן. בשפת BASIC זו אפשר לכנות משתנה בשם FRAN אם יש בכך כדי לסייע, אך עליך לנהוג משנה זהירות מכאן ואילך. רק שתי האותיות הראשונות נבדקות, כך שכל שם שיכיל אותן שתי אותיות עלול לבלבל ולסבך. מוטב שלא תשתמש ב-TOTAL כלל וכלל, מלה המכילה את המלה השמורה TO והדבר עלול לגרום לשגיאת תחביר. לא תמיד ניתן לאתר בקלות מלים אלו וה-BASIC MITS בודקת זאת רק בעת ההרצה, חסרון ממשי של שפת ה-BASIC. אם אתה מתעתד לעבוד על תכנית ארוכה, נסה ראשית חכמה את שמות הנתונים!”

קטע מן המאמר –
"Altair BASIC" מאת:
קית' בריטון ובווב מאלן.



חסוך זמן!

חסוך מקום!

לפני שנתקדם לתכניות ארוכות יותר ולעוד הוראות בשפת BASIC, נביא כאן מספר שיטות בהן ניתן לחסוך זמן, מקום והדפסה. חיסכון שנהגנו לכל אורך הדרך הוא להשמיט את הוראות ה-END ב-BASIC. בגירסאות הראשונות של BASIC, הייתה חובה לסיים כל תכנית בהוראת END כהוראה האחרונה (עם המספר הסידורי הגבוה ביותר). זה שוב אינו נחוץ בגירסאות החדשות המשוכללות, כך שאפילו לא הזכרנו זאת כאן. אך אין זה כל מה שאתה יכול להשמיט. נסה את תכניות התרגול הבאות.

999 END



הבט, אין
כאן LET!



NEW

```
OK
10 LET M=5
20 Q=10
30 PRINT M*Q
RUN
50
OK
```

היתכן? האם הערכים אכן הוצבו למשתנים? הדפס את שתי ההוראות בגישה ישירה, כדי לוודא שהערכים הוצבו עם או בלי LET בעת ההרצה.

PRINT M

5

OK

PRINT Q

10

OK

M	5
Q	10



אם המחשב שלך חישב $M*Q$ כ-5 פעמים 10, אתה יודע שהוא הציב את הערך 10 למשתנה Q למרות שהשמטנו את הוראת ה-LET עתה, כשאתה כבר מבין שאינך זקוק ל-LET בהוראת ה-LET, אתה מאפשר למשתנה Q לקבל את הערך 10, ולעולם אינך צריך להשתמש שוב ב-LET בתוך הוראת ה-LET (ועתה חוזר על כך מלולית - כן, מלה במלה!).

כשהינך משתמש ב-LET אתה גוזל מקום בזיכרון של המחשב, אך אתה חוסך מעט זמן בכך שהינך משיג ביצוע מהיר יותר עלידי המחשב.



נסה עוד הוראה נטרלת LET...

NEW

```
OK
10 G=13
20 PRINT G
RUN
13
```

OK

G	13
---	----



וכך למען היעילות והנוחות, הוגלה ה-LET לעולם ועד מארץ ההוראות.

המנע מקשיחות לב מיותרת!



השמטת ה-LET אינה הקיצור היחיד שהנהיגה גירסת ה-BASIC החדשה. היא גם דואגת להקשחת האצבעות בהן נוהג המתכנת להדפיס. הנה דרך מקוצרת להכנסת הוראת ה-PRINT:

NEW

OK

10 PRINT "HELLO GOOD LOOKING"

20 ? "USING ? FOR PRINT"

RUN

HELLO GOOD LOOKING

USING ? FOR PRINT

OK

פירושו PRINT בשעת הקלט

ועתה הדפס את התכנית ורשום מה מחליף את סימן השאלה בשורה 20.

LIST

10 PRINT "HELLO GOOD LOOKING"

20 PRINT "USING ? FOR PRINT"

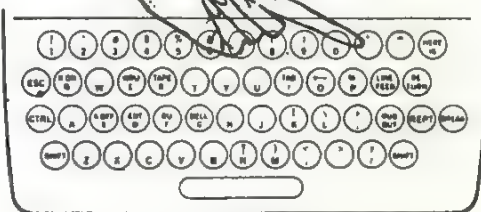
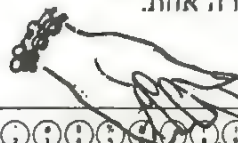
OK

אתה משתמש ב-? כשהינך מכניס התכנית (חיסכון בהדפסה). אך כשהינך מקיש על הקליד LIST, מדפיס המחשב החכם PRINT.



תכניתנו הקצרה הבאה מוכיחה שאתה יכול לדחוס ולצמצם את ההוראות שהינך נותן למחשב ולהגיע לפחות הוראות, באמצעות הכנסת מספר הוראות לשורה אחת.

השתמש ב-(: (נקודתיים) כדי להפריד בין ההוראות בשורה המכילה יותר מהוראה אחת.



מקלדת

NEW

OK

10 A=5 : B=10 : C=15 : D=20

20 ? A+B+C+D

RUN

50

OK

LIST

10 A=5 : B=10 : C=15 : D=20

20 PRINT A+B+C+D

OK

4 הוראות נטולות LET בשורה אחת!

צורת ה-? כתחליף ל-PRINT, זוכר?

? הופך לפני עיניך ל-PRINT!



בצע

טוב, אתה יכול הכניס לשורה כמה הוראות שתמצא, תוך כדי שימוש ב-:, להפריד בין ההוראות השונות

ריבוי הוראות



כתוב את התכנית שבעמוד הקודם, אך הכנס את כולה לשורה מרובת הוראות אחת.

NEW

OK

10 W=2 : X=4 : Y=6 : Z=8 : ? W*X*Y*Z

RUN

384



רואה את הוראת ה-PRINT בסוף השורה?

OK

LIST

10 W=2 : X=4 : Y=6 : Z=8 : PRINT W*X*Y*Z

OK

כמובן שזה פועל גם עם משתני מחרוזות.

NEW

OK

10 PS="YES" : QS="NO" : RS="MAYBE" : ? PS, QS, RS

RUN

YES

NO

MAYBE

OK

LIST

10 PS="YES" : QS="NO" : RS="MAYBE" : PRINT PS, QS, RS

OK



זו דרך דחוסה משהו לכתיבת תכנית, וייתכן שתמצאו לכתוב את תכניתיך בלי ריבוי הוראות בשורה אחת, כדי לשמור על בהירות. מאוחר יותר, תוכל לכתוב התכנית מחדש, בצפיפות גדולה יותר, אם יתעורר קושי כלשהו עם האיחסון במחשב שלך (אם זו תכנית ארוכה ומסועפת, המורכבת ממחרוזות ומנתונים רבים). לעת עתה, אתה יכול לחסוך בהדפסה.

אך זהירות, יש מספר הגבלות לאפשרויות שלך בשורה מרובת הוראות. GOTO עשוי להופיע רק כהוראה האחרונה בשורה מרובת הוראות, אך בשום אופן לא באמצע אותה שורה.

ייתכן שלהוראת DATA אין הוראות אחרות לפני או אחרי ה-DATA, ועל כן אסור להשתמש בהוראת DATA כשורה מרובת הוראות. הוראות DATA עומדות תמיד בפני עצמן.

רישום הערה-REM

והנה הוראה נוספת, החייבת לעמוד בפני עצמה, או להיות האחרונה בשורה מרובת הוראות, הלוא היא הוראת ה-REM (שורת הערה).



NEW

OK
10 REM-SET X EQUAL TO 5 : X=5
20 ? X
RUN

המחשב לעולם אינו רואה $x=5$, כי כשהוא מגיע ל-REMARK הוא מדלג אוטומאטית לשורה חדשה בתכנית.

0 ← הואיל ול- x לא הוצב כל ערך, מניח המחשב ש- $x=0$ ולכן הוא מדפיס 0. האם ידעת זאת? משתנה בלתי מוגדר שווה לאפס.

NEW

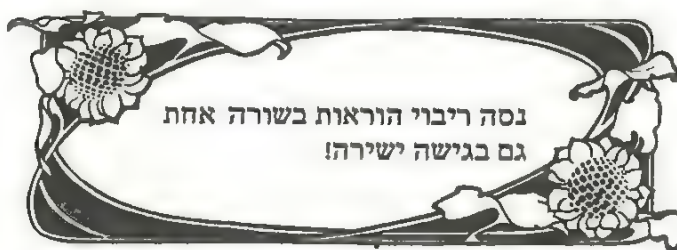
OK
10 X=5 : REM-SET X EQUAL TO 5
20 ? X
RUN

5 ← הפעם מבצע המחשב $x=5$ עוד בטרם הוא מבחין ב-REM, ואז הוא עובר לשורה הבאה בתכנית ומדפיס את הערך המוצב ל- x .



קרא

הוראות ה-REM מיועדות לאדם הקורא (או הכותב!) תכנית, הן משמשות להסברת משמעותה של שורה מסוימת או קבוצת שורות מתוך תכנית. לעתים קרובות הן גם משמשות לציון התחלת תת-תכנית (סוברוטין) שהיא קטע מתכנית, המורכב משורה אחת או יותר, המבצע חלק ממשימת התכנית כולה. בהמשך, כשנגיע לתכניות ארוכות ומורכבות יותר עוד תשמע על תת-תכניות. בינתיים, כשתרצה להסביר ולבאר את תכניתך לקורא, הסתפק בהוראת REM.



נסה ריבוי הוראות בשורה אחת
גם בגישה ישירה!

מאפשרת למתכנת להכניס הערות או ביאורים בגוף התכנית. הוראה זו אינה הוראה לביצוע.

מס' הערות
סידורי REM

10 REM THIS PROGRAM WAS DESIGNED BY MJMC
20 REM VARIABLES USED X, Y, Z, A, B, C

רק בגירסת BASIC PLUS

סימן קריאה, ! מפריד בין החלק שיש לבצע לבין ההערה באותה שורה.

140 LET Z = P*Q ! CALCULATES Z

ריבוי הוראות בשורה אחת

מופרדות על ידי נקודתיים (:).

רק בגירסת BASIC PLUS

השתמש ב-(:) או ב-חזור אות

פרק 4: בעיות

1. הקף בעיגול את אותו חלק, ההופך את שמות המשתנים שבהמשך ל"בלתי חוקיים".

CLEARANCE	REMARKABLE	HIGHSTEP
DIMENSION	SINNER	FASINATE
FOREIGN	ABSOLUTELY	FANTABULOUS
CARDIFF	INTERESTING	READABLE
POSITIVE	MANTAN	HOLDOUT
FREEDOM	DEFINATE	GLUCOSE
AMPERSAND	PEEKABOO	

2. בשורת ריבוי הוראות, אפשר להכניס הוראת DATA רק

3. בשורת ריבוי הוראות, אפשר להכניס הוראת GOTO רק כ-

4. נסה לחזור על בעיה 5 בסיומו של פרק 3, כתכנית המורכבת משורה מרובת הוראות.

5. כתוב מחדש שתיים או שלוש תכניות שכתבת בפרקים הקודמים כשאתה מנצל לטובתך את כל אותם תכסיסים לחסכון בזמן ובמקום.

חטא ההשמטות וטעויות אחרות שלי

מלה מן המחבר הנחמד שלכם:

כל הרווחים שאני מותיר בהוראת ה-BASIC אינן דרושים כלל וכלל: אין צורך ברווח אחרי מספר סידורי, ולא אחרי הוראות BASIC כדוגמת PRINT, READ, DATA, INPUT והוראות אחרות וגם לא אחרי ערכיה של הוראת PRINT (אחרי פסיקים או נקודות פסיק). הרווחים אינם נחוצים גם בין איבריה השונים של הוראת ה-DATA כל רווח מכביד מעט על זיכרונו של המחשב שלך, הדבר מהווה בעיה רק במקרה שתכניתך ארוכות מאוד ומכילות ערכי DATA רבים ומחרוזות רבות. אני מכניס רווחים אלה רק כדי להקל עליך להבין מה קורה בכל הוראה והוראה. זכור, אם ברצונך לחסוך בהדפסה, אתה יכול לוותר על הוראות ה-REM.

ב-BASIC בגירסת ALTAIR אינך חייב לתת ערכים למשנים שלך דהיינו, להתאים את ערכי המשתנים שלך לערכים שהוצבו להם בתחילה. אם אינך מבצע זאת אזי מציב ה-BASIC בגירסת ALTAIR אוטומאטית את הערך אפס לאותו משתנה, בפעם הראשונה שהוא פוגש בו בעת הרצת התכנית. זהו תהליך בלתי נראה, כלומר הוא אינו מופיע בתכנית עצמה. אם אתה מחלק את תכניותיך עם אנשים אחרים, רצוי להתאים את המשתנים של תכניתך במקום להניח למחשב לעשות זאת. כך תמנע מבוכה ובלבול מחבר המשתמש גם הוא בתכניתך, או אפילו מעצמך במקרה ולא תשתמש בתכנית מיד לאחר כתיבתה.

הסתמך על הסברים וחוקים קודמים של BASIC כדי שיסייעו לך בכתיבת

תכניות ובהרצתן בלי טעויות ותקלות.

תוכל להבחין בתכניות שלא הצלחת להדפיס בניסיון הראשון. ה-? הוחלף בהן ל-PRINT ובראשון הוצב NEW/OK. אל תתן לעצמך להתיאש בשל ההדפסה. מתסכל למדי להגיע לשורה מסוככת של ריבוי הוראות ואז לגלות שגיאה בתחילתה. השגיאות התכופות שאני עושה, הן: -

1. שוכח את הנקודתיים של ריבוי ההוראות, ביחוד אחרי הוראת PRINT. מסתיימת בנקודה פסיק או בפסיק.
2. שוכח את ? עבור PRINT, במיוחד בגישה הישירה.
3. משתמש בקליד SHIFT כשאינן צורך בכך, או לא משתמש ב-SHIFT כשיש בכך צורך.
4. שוכח את המרכאות בסופה של שורה ארוכה, בעיקר בתכנית הוראות משחק, היכן שהמשפט אינו מסתיים בסופה של שורת ההוראה PRINT. עליך להדפיס שוב את השורה כולה, רק כדי להכניס את המרכאות בסופה.
5. שוכח חלק אחד של הסוגריים.
6. ועוד שגיאה מתסכלת. הכנסת הוראה תוך שימוש מוטעה במספר הסיידורי של הוראה שכבר הכנסת קודם. ההוראה הראשונה הנורשאת מספר זה מוחלפת בשורה השניה. צרה כפולה ומכופלת: עליך להכניס מחדש את שתי השורות. חדד את חושיך. הצירוף SHIFT/@ (ראה עמוד 14) הוא זה שימנע את מחיקת השורה שהודפסה קודם (כמובן, ששורה זו היא תמיד השורה המסוככת ביותר בתכנית). אך עליך להבחין בטעות לפני שהינך מקיש על RETURN.

מותר לנו כמובן לפצל את שורות ריבוי ההוראות שלנו ואת הוראות הגישה הישירה להוראות בודדות. הואיל ואנו ממספרים את תכניותינו בעשרות, שורה בת שלוש הוראות שמספרה 10, יכולה להתפצל לשורות 10, 11 ו-12. אתה עשוי לחסוך בזמן ובמאמץ בהרצה הארוכה.

תזכורת למשתמשים בגירסת DEC BASIC PLUS ובגירסאות אחרות של BASIC: התכניות המודפסות וההרצות בספר זה נעשו ב-Altair 8K BASIC, גירסא 2.3. ייתכן שיש שוני מסוים בהודעה על שגיאות ובפרטים מסוימים בין גירסאות BASIC השונות. בדוק את רשימת השגיאות במערכת שלך למניעת בלבול. כוודאי תמצא את רשימת השגיאות בנספח לחוברת ההפעלה המצורפת למערכת המחשב שלך, כמובן אם היא לא נעלמה... בספר זה אנו מלמדים אותך כיצד לתת למחשב הוראות ב-BASIC. איננו מלמדים אתכם כיצד מתמודד המחשב עם סוג זה של הוראות, או כיצד הוא מצבע את תפקידיו. אם יש לך יותר מעניין חולף בנושא המחשבים, קרא את ספרו של ג'ון וייט, "המחשב הביתי שלך".

COMUTER.

בספר זה נראה לך מהן הטעויות הנפוצות ביותר בתכנות. אתה למד גם מטעויות. כמו כן תלמד מהם הגבולות לשימוש ב-BASIC. בנוסף על כך, תתוודע להודעות השגיאה המשוברות על ידי BASIC. הודעות אלו מסייעות לך באבחנת השגיאות בתכניתך ובתיקונן.

השווה והחליט



ועתה אנו מגיעים לאחת מסגולותיה החשובות והמענינות של שפת ה-BASIC: היכולת להשוות ולהחליט. בעגת מחשבים מסורתית, נקרא מושג זה בשם הפנייה מותנית. זהו כינוי למה שקורה כשהמחשב ממלא הוראה השולחת אותו לחלק אחר של התכנית, לא לפי הסדר המספרי של השורות. אתה עשוי להתייחס גם אל GOTO, כאל הפנייה, מפני שהיא מורה למחשב לפנות להוראה מסוימת ולבצע אותה, במקום להמשיך לפי סדרן המספרי של השורות בתכנית.

GO TO (מספר שורה)

השורה אליה צריך לפנות.

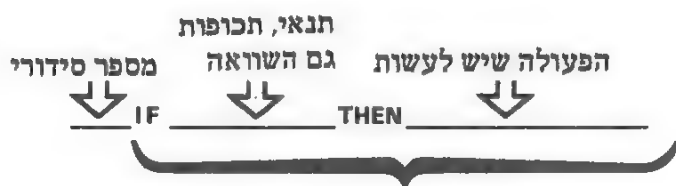
נא להכיר

IF... THEN אם... אז

אנו חותרים כאן אל שורש ההפנייה המותנית. אם תחשוב על כך, תבין שאם הפנייה פירושה GOTO, הרי שהפנייה מותנית פירושה GOTO בתנאים מסוימים בלבד. התנאי הוא המקום בו על המחשב להשוות ולהחליט. התנאי יכול להיות נכון או לא נכון. המחשב צריך להשוות ולהחליט באם התנאי נכון, או לא נכון. וכעת, אנו מגיעים סוף סוף להוראת ה-BASIC המורה למחשב להשוות ולהחליט: הוראת ה-IF-THEN. למעשה יש משפחה שלמה של הוראות IF-THEN. ממבט ראשון, זוהי הוראת ה-IF-THEN בצורתה הכללית ביותר.



אין להחליף את הסתעפות ענפיו של עץ בהסתעפות המותנית - IF... THEN



כל זה מהווה הוראת BASIC אחת.

בתנאי אחד



אזהרה ראשונה: אם מסתבר שהתנאי אינו נכון, עובר המחשב לביצוע ההוראה הממוספרת הבאה. פירוש הדבר שאם אתה מכניס הוראת IF...THEN בשורת ריבוי הוראות, המחשב לא יראה ולא יבצע הוראות המופיעות אחרי הוראת ה-IF...THEN, במקרה שהתנאי אינו נכון. אך אם התנאי נכון, יעבור המחשב לביצוע הפעולה הבאה בעקבות THEN, וימשיך בביצוע כל שאר ההוראות המופיעות בשורת ריבוי ההוראות. כפי שתראה, יש בכך תועלת מרובה.

כבר אמרנו שקיימת משפחה שלמה של הוראות IF...THEN. התנאי (אותו חלק, בין IF ל-THEN) מופיע בשישה טעמים נהדרים, שתראו בתוך המסגרת שבעמוד הבא. כל בני משפחת ה-IF...THEN, יכולים להכניס רק אחד מאותם שישה תנאים שונים בין ה-IF ל-THEN, אך לא זה מה שמבדיל בין בני המשפחה. ההבדל הוא במה שבא אחרי THEN, כלומר איזו פעולה יש לבצע אם התנאי אכן נכון.

הוראת ה-IF...THEN המקורית, ההוראה הנפוצה ביותר בהפנייה המתנית, היא כאילו היה לנו GOTO (מספר סידורי) אחרי THEN. למעשה, אתה יכול להכניס זאת כך:

IF (תנאי) THEN GO TO (מספר סידורי)

אפילו בשפת ה-BASIC המיושנת היית כותב זאת כך, בלי GO TO (כמו השמטת LET בהוראה המציבה ערך למשתנה, אתה כבר יודע...).

IF (תנאי) THEN (מספר סידורי)

↑
ה-GOTO מובן,
אך אינך חייב
להכניס אותו כאן.

להשוואה יש שתי תוצאות אפשריות. ההשוואה יכולה להיות נכונה או לא נכונה. בואו נתבונן בשני המקרים האלה: ההשוואה נכונה, ההשוואה לא נכונה.

לא נכון

IF (תנאי נכון) THEN (פעולה שיש לבצע)

↓
אזי שוכח המחשב
חלק זה ועובר
להוראה הממוספרת
הבאה בתכנית.
... (פירוש הדבר, כמובן, שהוא שקר) ...

נכון

IF (תנאי נכון) THEN (פעולה שיש לבצע)

➔ המחשב מבצע מה
שנאמר לו לעשות
בחלק זה, כאשר
התנאי נכון.
המחשב משווה כאן
ומחליט שהתנאי
נכון.

6 תנאים אפשריים

IF...THEN

אם הערך הראשון קטן מן השני	<	<
אם הערך הראשון גדול מן השני	>	>
אם שני הערכים שווים	=	=
אם הערך הראשון נמוך מהשני או שווה לו	(< או) =	<=
אם הערך הראשון גדול מהשני או שווה לו	(> או) =	>=
אם שני הערכים אינם שווים	(< או) #	<>

אם אינך מכיר סימולים אלה, כדאי שתתוודע אליהם עתה, כדי למנוע בילבול בעתיד.

IF...THEN PRINT



בת שניה למשפחת ה-IF...THEN היא הוראת ה-IF...THEN PRINT במקום ציון מספר סידורי, אחרי THEN, שאליו צריך המחשב לעבור, אתה יכול להכניס הוראת PRINT שתדפיס כל מה שהוראת PRINT יכולה להדפיס, הכול בהתאם לחוקים התואמים את הוראות ה-PRINT הרגילות. הוראת ה-IF...THEN PRINT תדפיס כמובן רק במקרה שהתנאי נכון.

(כל מה שהוראת PRINT יכולה לעשות) THEN PRINT (תנאי) IF

במקרה שהתנאי אינו נכון, המחשב לא ידפיס דבר, אלא יעבור הישר אל ההוראה הממוספרת הבאה בתכנית.

האם עייפת מן התאוריה ולבך יוצא אל המעשה? האם אתה מבובל? אל תתיאש, עוד תראה את האור בקצה המנהרה. בהדגמה הראשונה שלנו נשתמש בהוראת ה-IF...THEN PRINT, רק מפני שכך ניתן להמחיש טוב יותר את פעולת ההשוואה והבחירה של המחשב. נמחיש את הוראות ה-IF...THEN האחרות בבוא הזמן.



ריבוי ההוראות בשורה 20 נהנות מהעובדה שכשהתנאי ב-IF...THEN או ב-IF...THEN PRINT אינו נכון, הדבר גורם למחשב לדלג כלפי מטה אל ההוראה שבשורה הממוספרת הבאה. דהיינו, המחשב מתעלם מכל ההוראות המופיעות אחרי IF...THEN השקרי, המצויות באותה שורה. אך אם התנאי IF...THEN נכון, מבוצע גם המשך הוראת ה-IF...THEN וכמו כן כל ההוראות. תפסת? זהו תכסיס נוסף לשימוש בריבוי הוראות והתאמתן לצורכי התכנות שלך בעתיד.

GO TO לשורה 10
המשך להריץ את
התכנית.

אם התנאי נכון, הדפס זאת ועבור להוראה הבאה בשורה זו.

20 IF X<0 THEN PRINT "CONDITION TRUE:"; X; "< 0" : GOTO 10

אם התנאי לא נכון, המשך להוראה
בשורה הממוספרת הבאה, ושכח את
הכתוב בהמשך השורה.

בצע

NEW

OK

```
5 REM-COMPARE AND DECIDE IF X IS LESS THAN ZERO
10 READ X
20 IF X<0 THEN PRINT "CONDITION TRUE:"; X; "< 0" : GOTO 10
30 PRINT "CONDITION FALSE:"; X; "IS NOT < 0" : GOTO 10
40 DATA 4, 0, -3, 6, -2, 7, 9, -12
RUN
CONDITION FALSE: 4 IS NOT < 0
CONDITION FALSE: 0 IS NOT < 0
CONDITION TRUE:-3 < 0
CONDITION FALSE: 6 IS NOT < 0
CONDITION TRUE:-2 < 0
CONDITION FALSE: 7 IS NOT < 0
CONDITION FALSE: 9 IS NOT < 0
CONDITION TRUE:-12 < 0
```

70D ERROR IN 10 המחשב שלח לך הודעת
OK שפירושה OUT OF DATA
אך אל תתעצב אל לבך.



השתמש ב- וב- <



השתמש ב- SHIFT וב- > בעת ובעונה אחת

```
5 REM-COMPARE AND DECIDE IF X IS EQUAL TO OR GREATER THAN ZERO
10 READ X
20 IF X=>0 THEN PRINT "CONDITION TRUE:"; X; "=> 0" : GOTO 10
30 PRINT "CONDITION FALSE:"; X; "IS NOT => 0" : GOTO 10
40 DATA 4, 0, -3, 6, -2, 7, 9, -12
RUN
CONDITION TRUE: 4 => 0
CONDITION TRUE: 0 => 0
CONDITION FALSE:-3 IS NOT => 0
CONDITION TRUE: 6 => 0
CONDITION FALSE:-2 IS NOT => 0
CONDITION TRUE: 7 => 0
CONDITION TRUE: 9 => 0
CONDITION FALSE:-12 IS NOT => 0
```

70D ERROR IN 10
OK

אותה הודעה אומרת שהמחשב ניסה לקרוא (READ) ערך אחר עבור X, אחר שכל הערכים בהוראת ה-DATA היו בשימוש. למעשה, אנו יכולים להפוך זאת לדרך נוחה לעצירת המחשב לאחר ביצוע עבודתו.

חוקי התנאים - מה ניתן להשוואה?



כבר ראינו כיצד פועלת הוראת IF ...THEN באופן כללי. ועתה, הבה נתבונן רק בהשוואה או בהתנייה עצמה ונראה מה עומד להשוואה.

NEW

```
OK
10 X=5 : Y=20
20 IF 2*X=Y/2 THEN ? "TRUE"
RUN
TRUE
```

OK

20 IF 2*X=Y/2 THEN ? "TRUE"



שני הפריטים העומדים להשוואה יכולים להיות משתנים, או ביטויים שיש לחשב. גם שתי מחרוזות ניתנות להשוואה.

הדרך המקוצרת להכנסת PRINT, על-ידי סימן השאלה, פועלת גם עבור המלה PRINT בתוך IF ...THEN PRINT ההוראה

השוואת מחרוזות



NEW

משפחת ההוראות IF ...THEN מסוגלת גם להשוות בין מחרוזות. לצורך המחשה, נשתמש בהשוואה פשוטה: האם מחרוזת הנקראת מהוראת ה-DATA שווה למחרוזת "YES"? אם התשובה חיובית, אזי ההשוואה נכונה. בהמשך תלמד כיצד להשתמש בחמש ההשוואות האחרות בין מחרוזות. לעת עתה, די אם תנסה את התכנית שבהמשך.

OK

```
5 REM-COMPARE AND DECIDE IF X$ STRING IS EQUAL TO "YES"
10 READ X$
20 IF X$="YES" THEN PRINT "CONDITION TRUE:"; X$; " = YES" : GOTO 10
30 PRINT "CONDITION FALSE:"; X$; " DOES NOT = YES" : GOTO 10
40 DATA YES, NO, MAYBE, YES, NO, MAYBE
```

RUN

```
CONDITION TRUE:YES = YES
CONDITION FALSE:NO DOES NOT = YES
CONDITION FALSE:MAYBE DOES NOT = YES
CONDITION TRUE:YES = YES
CONDITION FALSE:NO DOES NOT = YES
CONDITION FALSE:MAYBE DOES NOT = YES
```

```
?00 ERROR IN 10
OK
```

20 IF X\$="YES" THEN PRINT



שים לב לכך שהמחרוזת חייבת להופיע בין מרכאות.

בחן והחלט



בתכנית הבאה כלולות שלוש הוראות IF ...THEN PRINT לבחינת אותו ערך, עד שלוש פעמים. יש שלושה תנאים שונים העומדים לבחינה. שים לב לריבוי ההוראות בשורות 20, 30 ו-40. אם התנאי IF ...THEN PRINT אינו נכון, עובר המחשב לשורת ההוראה הממוספרת הבאה, ומתעלם מכל שאר ההוראות שאחרי ה-IF ...THEN. IF השקרי באותה שורה, אך אם התנאי - IF ...THEN PRINT נכון, אזי מבוצעת כל הוראת ה-IF ...THEN PRINT, כמו כל ההוראות האחרות שבאותה שורה.



NEW

OK

```
5 REM- NEGATIVE, POSITIVE AND ZERO NUMBER TESTER
10 INPUT "INPUT 0 (ZERO) OR ANY NEGATIVE OR POSITIVE NUMBER"; N
20 IF N<0 THEN ? "YOUR NUMBER IS NEGATIVE." : ? : GOTO 10
30 IF N>0 THEN ? "YOUR NUMBER IS POSITIVE." : ? : GOTO 10
40 IF N=0 THEN ? "YOUR NUMBER IS ZERO." : ? : GOTO 10
RUN
INPUT 0 (ZERO) OR ANY NEGATIVE OR POSITIVE NUMBER? 8975
YOUR NUMBER IS POSITIVE.
```

```
INPUT 0 (ZERO) OR ANY NEGATIVE OR POSITIVE NUMBER? -384
YOUR NUMBER IS NEGATIVE.
```

```
INPUT 0 (ZERO) OR ANY NEGATIVE OR POSITIVE NUMBER? 0
YOUR NUMBER IS ZERO.
```

```
INPUT 0 (ZERO) OR ANY NEGATIVE OR POSITIVE NUMBER?
```

OK

הבה נתבונן מקרוב באחת מהוראות IF ...THEN PRINT.

הדפס שורה ריקה
(בין הלולאות ל-
(RUN)

```
20 IF N<0 THEN ? "YOUR NUMBER IS NEGATIVE." : ? : GOTO 10
```

אם התנאי אינו נכון, המשיך לשורת ההוראה הממוספרת הבאה, ושכח את המשיך השורה. פירוש הדבר שאינך עושה דבר אחרי ה-THEN.

GOTO לשורה 10
והמשיך להריץ את התכנית.

מהו מסנן?



התכנית הבאה דומה לתכנית הקודמת, אך עליך לשים לב לכך שהתנאים הנבחרים הם השוואה בין שני ערכי ה- X ו- Y . כוג זה של תת-תכנית משמש בתכניות רבות לבדיקת דברים שונים (עריכת השוואות רבות) בקשר לנתונים ולקלט. זו כמעט תכנית משחק.



NEW

OK

```
5 REM-COMPARISON OR "FILTER" PROGRAM
10 INPUT "INPUT ANY TWO NUMBERS"; Y,Z
20 IF Y<Z THEN ? Y; "IS LESS THAN "; Z
30 IF Y>Z THEN ? Y; "IS GREATER THAN "; Z
40 IF Y=Z THEN ? Y; "IS EQUAL TO "; Z
50 ?
60 GOTO 10
```

אנו גורמים למחשב לתבוע שני משתנים, ולהציב אחד ל- Y ואחר השני ל- X .

RUN

INPUT ANY TWO NUMBERS? 8,53
8 IS LESS THAN 53

INPUT ANY TWO NUMBERS? 2001,1999
2001 IS GREATER THAN 1999

INPUT ANY TWO NUMBERS? -12,-33
-12 IS GREATER THAN -33

התיקון עלידי → הוא OK כשמכניסים INPUT, אך רק אחרי שהקשת על RETURN...

INPUT ANY TWO NUMBERS? 81,81
81 IS EQUAL TO 81

INPUT ANY TWO NUMBERS?
OK

לחץ על RETURN כדי לעצור את הרצת התכנית ולהמתין ל-INPUT אם הדבר אינו פועל, נסה CONTROL/C.



האם זכרת להפריד את ערכי ה-INPUT שלך עלידי פסיק, אחרי סימן השאלה של INPUT?

שים לב לכך שהמחשב משווה את ערכי Y ו- Z שלוש פעמים, בין אם הוא חייב לעשות זאת אם לא. השווה זאת ל-NEG, POS, ZERO NUMBER TESTER.

בתוך המסנן (המסנן את האפשרויות השונות, תפסת?) מבצע המחשב, בחלקה של התכנית (שורות 20, 30, 40) רק את הנדרש ממנו עלידי הוראת IF (דהיינו להדפיס) במקרה שהשוואה נכונה.

להדפיס או לא להדפיס:

זאת השאלה



ועתה הקדש תשומת לב רבה לתכניות הקטנות שלהלן, המשמשות בהוראות IF...THEN שורה 20 משמשת כדי להחליט אם להדפיס את הערך או לא. במקרה שהתנאי שקרי, עובר המחשב להוראת ה-PRINT שבשורה 30 ומדפיס את ערכו של R. אך כשהתנאי נכון, המחשב מסתעף חזרה לשורה 10. הוא אינו מגיע לשורה 30 ולכן אף אינו מדפיס. תחת זאת, מוצב ערך חדש ל-R מתוך ה-DATA. הערך החדש נבחן עלידי שורה 20. וכך הלאה. בתחילה, עשוי הדבר להיראות לך בסדר הפוך. עליך לזכור שהפעולה המופיעה אחרי THEN מבוצעת רק במקרה שהתנאי נכון. ועתה שנס מותניך ונסה את התכניות.



NEW

```
OK
5 REM-DON'T PRINT NUMBERS LESS THAN ZERO
10 READ R
20 IF R<0 THEN 10
30 ? "R =" R
40 GOTO 10
50 DATA 4, 0, -3, 6, -2, 7, 9, -12
RUN
R = 4
R = 0
R = 6
R = 7
R = 9
```



בדוק את השיחזור בעמוד הבא. ➡

⬅ 70D ERROR IN 10 המחשב שיגר שוב הודעת OD. גם הפעם אין לך מה לדאוג.
OK

ועתה שנה את התנאי בהחלפת שורה 20. (השמט את ה-REM אם רצונך בכך).

```
5 REM-DON'T PRINT NUMBERS GREATER THAN ZERO
20 IF R>0 THEN 10
RUN
R = 0
R = -3
R = -2
R = -12
```



⬅ 70D ERROR IN 10 אותו מסר של המחשב, האומר שהמחשב ניסה לקרוא ערך ל-R אחרי שכל הערכים בהוראת ה-DATA היו בשימוש פעם אחת.

השיחזור להפנייה מותנית

להלן מובא שיחזור ההרצה של דוגמת IF...THEN הראשונה, המופיעה בעמוד האחרון. התנאי הנבחר כאן הוא $R < 0$ (קטן מ-0). התוצאה הינה "אל תריץ מספרים קטנים מאפס." עבור על השיחזור כדי להבין את ההחלטות שנתקבלו על-ידי המחשב: להדפיס או לא להדפיס.

הוראה	משתנים וערכים	תאור
5 REM-DON'T PRINT NUMBERS LESS THAN ZERO		המחשב מדלג על ההערות.
10 READ R	R 4	פריט ראשון משורה DATA 50 מוצב ל-R משתנה READ.
20 IF R<0 THEN 10	R 4	המחשב משווה את ערכו של R ל-0 (אפס). הואיל ו-4 אינו קטן מ-0, התנאי שקרי והמחשב ממשיך לשורה הבאה.
30 PRINT "R ="; R	R 4	ערכו של R מודפס.
40 GOTO 10	R 4	המחשב חוזר לשורה 10.
10 READ R	R 0	הערך הבא מ-DATA 50 מוצב ל-R.
20 IF R<0 THEN 10	R 0	0 אינו קטן מ-0, כך שהתנאי שקרי, זזים אל השורה הבאה...
30 PRINT "R ="; R	R 0	ערכו של R מודפס.
40 GOTO 10	R 0	וחזרה לשורה 10...
10 READ R	R -3	הפריט השלישי מ-DATA 50.
20 IF R<0 THEN 10	R -3	טוב, הפעם ערכו של $R = -3$, שהוא פחות מ-0 (התנאי נכון), כך שהמחשב ממלא את שאר הדרישות המופיעות בהוראת ה-IF...THEN, וקופץ חזרה לשורה 10, כשהוא מדלג על הוראת ה-PRINT שבשורה 30 וכו'.
10 READ R	R 6	ערך חדש ל-R.
50 DATA 4, 0, -3, 6, -2, 7, 9, -12		שורה 10 קוראת את הנתונים בשורה 50.

טכנית, תהליך זה נקרא הפנייה מותנית. אם התנאי נכון, המחשב מופנה לקטע אחר של התכנית.

אם התנאי נכון, מבצע המחשב את מה שמורה לו המשך ההוראה. אך אם התנאי אינו נכון, מתעלם המחשב מהמשך הוראת ה-IF...THEN ועובר ישר לשורת ההוראה הממוספרת הבאה שבתכנית.

עצור את הלולאה!

זהו התנאי

מספר סידורי

30 IF T<10 THEN 10

בצע אם התנאי נכון. במקרה זה, עבור להוראה בעלת מספר סידורי זה.



המשך לשורת ההוראה הממוספרת הבאה, אם התנאי שקרי. פירוש הדבר שעליך לעבור להוראה הבאה, בעלת המספר הסידורי הנמוך ביותר, הגבוה מ-30.

ועתה הכנס והרץ תכנית זו כדי לעצור את הלולאה לאחר מספר פעמים, כשהוראת ה-IF ...THEN עורכת את הבדיקה.

NEW

OK

5 REM-SELF-STOPPING COUNTING PROGRAM

10 ? T

20 T=T+1

30 IF T<10 THEN 10

40 ? "NOW T ="; T; "SO I STOPPED MYSELF."

RUN

0

1

2

3

4

5

6

7

8

9

NOW T = 10 SO I STOPPED MYSELF.

OK



הוראת ה-IF שלנו (שורה 30 דלעיל) מציגה בפנינו שאלה עמוקה, "האם ערכו של T קטן מ-10?" כשהתשובה היא "אמת", אזי 10 מורה למחשב לפנות או לדלג חזרה לשורה 10. אך כשהתשובה לאותה שאלה מעמיקה היא "לא" (כשערכו של T מגיע לבסוף ל-10), מתעלם המחשב לחלוטין מהוראת ה-IF ומדלג אל השורה הבאה בתכנית (שורה 40).

ההוראה IF THEN (IF GOTO)

גורמת לתכנית לקפוץ או לפנות במקרה שהתנאי נכון. אם לא כן, התכנית ממשיכה לשורת ההוראה הממוספרת הבאה.

או הוראה (מספר סידורי) THEN (תנאי) IF מספר סידורי

```
10 IF X 10 THEN 50
10 IF X 10 THEN PRINT "NUMBER IS LESS THAN 10"
10 IF X 10 THEN PRINT "ERROR" : X = 0 : GO TO 50
10 IF AS = "YES" THEN PRINT "GOODBYE FOR NOW" : END
```

בעקבות IF GOTO חייבת להופיע מספר סידורי.

משפחת הוראות IF ...THEN

ניתן להשמיט את GOTO
IF...THEN (GOTO) IF...THEN PRINT או IF...THEN ?
ניתן להשמיט את LET
IF...THEN (LET) IF...THEN INPUT
IF...THEN READ
IF...THEN GOSUB
IF...THEN RETURN
IF...THEN STOP
IF...THEN END

רק ב-BASIC PLUS

ב-BASIC PLUS יש == שפירושו "בערך שווה ל-"



פרק 5: בעיות

1. בהוראה IF-THEN, כשתנאי שקרי, הרי שהתכנית

2. בהוראה IF X < > THEN 10, המבחן הוא בקביעה אם שני המשתנים

3. IF X > = THEN 100 מנסה לקבוע את ערכו של X

מערכו של Y.
4. להלן תכנית האמורה לעבור על רשימת נתונים ולהדפיס את כל המספרים
הגדולים מ-100. מה לקוי בתכנית זו?

10 READ A
20 IF A > 100 THEN 30
30 GOTO 10
40 DATA 12, 112, 30, 230, 400, 24

5. מחשב את חשבון המים שלך! זוכר את בעיה 8 בסוף פרק 2? ניסית לחשב
את צריכת המים החודשית שלך בגאלונים. ועתה לחלק הפחות נעים, אל
החשבון עצמו.
הנח שכל אחד משלם סכום קבוע של \$2.85 לחודש ושהמים מחושבים לפי
רגל מעוקב (בכל רגל מעוקב 7.5 גאלונים).
כתוב תכנית שתמחשב את חשבון המים שלך, כשאתה נדרש לשלם \$0.50
ל-100 רגל מעוקבים שצרכת. הכנס את הצריכה בגאלונים (אל תשכח
להוסיף את הסכום הקבוע). עבור צריכת 6,000 גאלונים התשובה היא
\$6.85, ל-10,000 גאלונים = \$9.52 ול-20,000 גאלונים = \$16.18.
6. כשאנו משתמשים בנתונים של בעיה 5, הבה נניח שחברת המים מתחילה
לקבוע את הצרכנים הגדולים עבור המים שהם צורכים מעבר למכסה
הרגילה. הבה נניח שלמשפחה בת ארבע נפשות, מוקצים 8000 גאלונים
לחודש והיא נדרשת לשלם \$0.50 ל-100 רגל מעוקב על 8000 הגאלונים
האלה (בתוספת הסכום הקבוע). התשלום על הצריכה העולה על 8000
גאלון הוא \$1.00 ל-100 רגל מעוקב. כתוב תכנית שתמחשב את חשבון
המים בשיטה זו (זה עשוי לקרות).
7. מכור זמן מחשב לאחד משכניך... לצורך הבידור והרווח! תזדקק לשעון
זמן ולתכנית מחשב שתבצע את החישובים. גבה \$2.00 על הזכות
להשתמש במחשב בתוספת \$0.05 לדקה - עבור 45 הדקות הראשונות
ו-0.03 (\$) לכל 100 דקות.
$$45 \times .05 = 2.25$$
$$(100 - 45) \times .03 = 1.65$$
$$2.00 = \text{דמי חיבור}$$
$$\$5.90 = \text{ס"ה}$$

8. רמז. ומה אם השכן משתמש ב-45 דקות? אל תשכח לתכנת גם
אפשרות זו.
השוואת מחירויות: יש לך רשימת לקוחות ענקית, המורכבת מ-4000
שמות, בהוראות DATA עם שמות ומספרי מיקוד וכל זאת בשתי מחירויות
נפרדות. (מדוע הצבנו את מספרי המיקוד במשתני מחירויות במקום
במספרים?).
DATA LISA STEWART, 94061

אתה עומד לעבור לאזור שכמספר המיקוד שלו הוא 94061. עבור על
השמות שבהוראות ה-DATA שלך, בחר והדפס רק את השמות שבמיקוד
94061, כך שתדע למי להתקשר כשתהיה באותו אזור.

מסעך פונקציות

#1



שפת ה-BASIC כוללת כמה פעולות "אוטומטיות" המכונות 'פונקציות'. לעתים הן דומות לנוסחאות או משוואות לביצוע מספר דברים למספרים ולמחרוזות. בעמוד 158 תמצא את רשימת הפונקציות. הפונקציה הראשונה שלנו ב-BASIC היא הפונקציה SQR (X). זוהי פונקציית השורש הריבועי. היא מורה למחשב למצוא את השורש הריבועי של כל ערך המצוי בתוך הסוגריים.

באור לריבועים ולשורשים ריבועיים

ריבוע של מספר הוא מספר זה כפול בעצמו. לדוגמא, $9 \text{ פעם } 9 = 81$. בדיוק כמו 9^2 (תשע בריבוע) או 9×9 בשפת ה-BASIC. 81 הוא הריבוע של 9.

זה פועל גם בכיוון ההפוך. השורש הריבועי של מספר הוא מספר אחר, שאם יוכפל בעצמו, יתן את המספר הראשון. לדוגמא, 9 הוא השורש הריבועי של 81 מפני ש $9 \times 9 = 81$. בספרי מתמטיקה כותבים זאת כך:

$$\sqrt{81} = 9$$

9 - גם הוא השורש הריבועי של 81, מפני ש $(-9) \times (-9) = 81$. למרות זאת, מחשבת שפת ה-BASIC רק את השורש הריבועי החיובי. תוכל על כן למצוא רק את השורש הריבועי של מספרים חיוביים ולא של מספרים שליליים.

כפי שתווכח, השורש הריבועי של מספר אינו בהכרח מספר שלם, כדוגמת 3, אלא לעתים תכופות יותר מספר בעל שבר עשרוני.

השתמש בגישה ישירה כדי לערוך ניסויים עם הפונקציה SQR(X).
נסה את הדוגמאות שלהלן ונסה את רעיונותיך שלך.



```
PRINT SQR(100)
10
```

OK

SQR(משתנה, ערך,
או ביטוי
שיש לחשב. **)**

```
? SQR(85)
9.21955
```

OK

```
? SQR(1), SQR(2), SQR(3), SQR(4)
1          1.41421          1.73205          2
```

OK

```
? SQR(5); SQR(6); SQR(7); SQR(8); SQR(9); SQR(10)
2.23607 2.44949 2.64575 2.82843 3 3.16228
```

OK

ועתה כשאתה עדיין משתמש בגישה ישירה, נסה למצוא בעזרת BASIC את
השורש SQR של מספר שלילי.

```
? SQR(-25)
```

```
?FC ERROR
```

OK

```
? SQR(100)
10
```

OK

```
? SQR(-100)
```

```
?FC ERROR
```

OK

ראה הוזהרת ה-BASIC אומרת לך שהינך מנצל
לרעה פונקציה.

"FC ERROR" פירושו "טעות בפונקציה" נראה
שה-BASIC אינה אוהבת לחשב את השורש הריבועי
של מספרים שליליים.

ועתה, נסה מספר חישובים הכוללים את הפונקציה SQR.

```
? 5*SQR(9)
15
```

OK

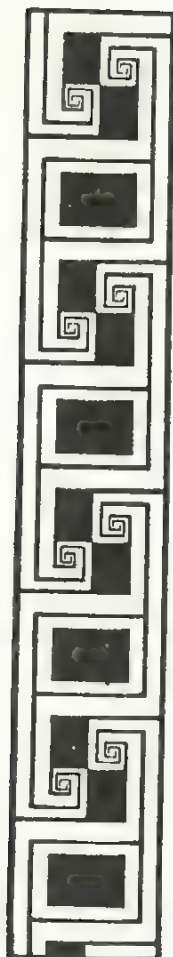
```
? 5*(10+SQR(9))
65
```

OK

אל תשכח לכתוב צמדים של סוגריים, שאם לא כן
תשלח לך ה-BASIC הודעה על SN ERROR.

```
X=7 : Z=SQR(X) : ? Z
2.64575
```

OK



ערוך את ניסוייך שלך ולאחר מכן נסה את התכניות הבאות:

NEW

```
OK
10 X=7
20 Z=SQR(X)
30 ? Z
RUN
2.64575
```



OK

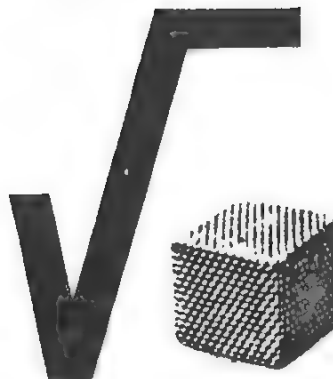
NEW

```
OK
10 INPUT A
20 ? A; SQR(A)
30 ? : GOTO 10
RUN
? 225
225 15
```

```
? 33
33 5.74456
```

? ← האם הקשת על RETURN

OK



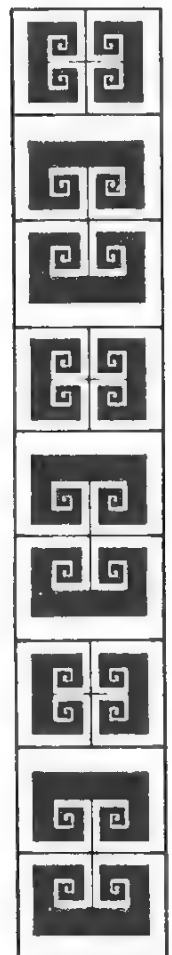
NEW

```
OK
5 REM-SQUARE ROOT MACHINE
10 ? "GIVE ME A NUMBER AND I WILL TELL YOU THE SQUARE ROOT."
20 INPUT "NUMBER PLEASE"; A
30 ? "THE SQUARE ROOT OF"; A; "IS"; SQR(A) : ? : GOTO 20
RUN
GIVE ME A NUMBER AND I WILL TELL YOU THE SQUARE ROOT.
NUMBER PLEASE? 7
THE SQUARE ROOT OF 7 IS 2.64575
```

```
NUMBER PLEASE? 1444
THE SQUARE ROOT OF 1444 IS 38
```

NUMBER PLEASE?

OK



INT()

לחישוב משתנה,
ערך, או ביטוי אלגברי.



הפונקציה $INT(X)$ מורה למחשב למצוא את השלם של מספר חיובי. השלם של מספר הוא המספר השלם, בלי השבר העשרוני, כדוגמת:

$$INT(5.6) = 5 \quad \text{השלם של 5.6 הוא 5}$$

מצא את השלם של המספרים שבסוגריים, בשיטת הגישה הישירה.



? $INT(3.1)$
3

OK

? $INT(3.9999), INT(101.1), INT(3.14159)$
3 101 3

OK

נסה עתה לעבוד עם מספרים שליליים. הפונקציה INT מחזירה את המספר השלילי בעל הערך המוחלט הגדול יותר. $int(-5.6) = -6$. השלם של -5.6 הוא -6.

? $INT(5.1), INT(-5.1), INT(5.9999), INT(-5.9999)$
5 -6 5 -6

OK

שים לב לפעולת המחשב
ביחס למספר שלילי,
כהשוואה למספר חיובי.

מצא את השלם של השורש הריבועי של 52.

? $SQR(52), INT(SQR(52))$
7.2111 7

OK



שים לב לשימוש המתוחכם של פונקציה בתוך הסוגריים של פונקציה אחרת. ושוב מתחיל ה-BASIC בחלק הפנימי ביותר של המרכאות. אל תשכח להתאים זוגות סוגריים.

עיגול מספרים באמצעות INT

לעתים נוח יותר לעגל חישוב מסוים מאשר פשוט לקצוץ את השבר העשרוני. עיגול מספר, משמעו שהמספר הופך לשלם, כשהשבר העשרוני הוא נמוך מ-5 (חצי), כלומר 5.4 הופך ל-5. אם השבר העשרוני של מספר הוא 5 או יותר, אזי מתעגל המספר כלפי השלם הגדול יותר, כלומר 5.6 מתעגל ל-6.

בדרך זו אתה משתמש בפונקציה INT כדי לעגל מספר.

$INT(x + .5)$ $INT(5.4 + .5) = 5$ (5.9 בלי ה-9)
 $INT(5.6 + .5) = 6$ (6.1 בלי ה-1)

הרץ תכנית זו כדי לעגל את המספרים החיוביים שבהוראת ה-DATA.



```

NEW
OK
5 REM-NUMBER ROUNDER OFFER
10 READ X : ? X, INT(X+.5) : GOTO 10
20 DATA 3.14159, 101.888, 521.999, 521.001, 521.5
RUN
3.14159      3
101.888      102
521.999      522
521.001      521
521.5        522
    
```

?OD ERROR IN 10 ← OUT OF DATA הודעת
OK

עיגול סכומי כסף

בעיה לדוגמא: חשב את מס הקנייה על פריט שמחירו \$17.95 ושיעור המס עליו הוא 6%. עגל את התוצאה עד הפני השלם הקרוב ביותר.

$$\$17.95 \times 0.6 = \$1.077$$

(1) הכפל את הסכום ב-100 כדי לעגל אותו. פעולה זו מזיזה את הנקודה העשרונית שני מקומות ימינה.

$$1.077 \times 100 = 107.7$$

(2) הוסף .5 למספר לצורך עיגול.

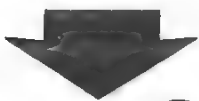
$$107.7 + .5 = 108.2$$

(3) קח את השלם של ערך זה.

$$INT(108.2) = 108$$

(4) חלק ל-100 כדי להזיז את הנקודה העשרונית חזרה למקומה המקורי, שני מקומות שמאלה.

$108/100 = 1.08$, כך שקיבלנו 1.08 כמס קנייה על \$17.95 מעוגל לפני השלם הקרוב ביותר.



ועתה, הבה ניתן למחשב לפתור בעיה זו. שים לב לכך שאנו מורים למחשב להדפיס את הערך של M אחרי כל שלב בתהליך העיגול, כדי לסייע לנו לראות את התרחשות התהליך, צעד אחר צעד.



NEW

OK

10 $M=17.95*.06$: ? M

20 $M=M*100$: ? M

30 $M=M+.5$: ? M

40 $M=INT(M)$: ? M

50 $M=M/100$: ? M

RUN

1.077 ←

הערך המחושב שיש לעגל.

107.7 ←

הנקודה העשרונית זוהי שני מקומות ימינה.

108.2 ←

הקונסנטה לעיגול, (.5) מתווספת.

108 ←

השבר העשרוני קוצץ.

1.08 ←

הנקודה העשרונית מוחזרת למקומה המכובד, שני מקומות שמאלה, ואנו מקבלים סכום

כסף עגול.

OK



ניתן לקבל אותה תוצאה בעזרת הוראה אחת ויחידה. (תוכל להשתמש בגישה ישירה בתור שורת הוראות אחת).



? $INT(17.95*.06*100+.5)/100$
1.08

1 חישוב המס

2 הזז את הנקודה העשרונית שני מקומות ימינה.

4 קח את השלם של תוצאת החישובים שבתוך הסוגריים

5 השלב האחרון הוא חלוקה ב-100 כדי להסיט את הנקודה העשרונית 2 מקומות שמאלה, מקום בו היא צריכה להיות.



זכור דוגמא זו בה תוכל להשתמש בתכניות המחשב הבאות שלך. השתמש בפונקציה INT לעיגול חישובים בדולרים.

NEW

OK

10 $M=17.95*.06$

20 ? $INT(M*100+.5)/100$

RUN

1.08

M עשוי להיות תוצאה של חישובים, או יכול להיות ערך שחושב בשלב מוקדם יותר באותה תכנית.



קרא

חילוק עם שארית

בוא נשתמש בפונקציה INT לפיתרון בעיות "חילוק עם שארית".

$$\begin{array}{r}
 \text{מנה} \leftarrow 19 \\
 \text{מחולק} \leftarrow 97 \\
 \text{מחלק} \leftarrow 5 \\
 \hline
 47 \\
 \hline
 45 \\
 \hline
 2 \leftarrow \text{שארית}
 \end{array}$$



בצע

ועתה הבא את המחשב לידי כך שיבצע זאת עבורך. ראשית, הנה הכעיה:
 A מחולק ל- B , A/B (או $B \overline{A}$) כש- $A = 97$ ו- $B = 5$.

NEW

OK

5 REM-LONG DIVISION CRANKER OUTER

10 A=97 : B=5 ← הערכים של A ו-B מוצבים בשורה זו.

20 Q=INT(A/B) ← המנה: מספרן השלם של הפעמים ש-B נכנס ל-A (5 נכנס ל-97).

30 R=A-B*Q ← שורה 30 מקבלת את השארית: מה נשאר מ-A כשהינך מחלק אותו ל-B? (97 - (5*19)).

40 ? "QUOTIENT =" ; Q : ? "REMAINDER =" ; R האם הבחנת בדרך המסודרת

בה ביצענו זאת?

QUOTIENT = 19

REMAINDER = 2

OK

↑ שים לב: שורה 20 מחשבת למעשה את התוצאה עם שבר עשרוני, המקוצץ על ידי INT.

נסה זאת עם הערכים שלך, עלייך החלפת שורה 10 והוספת שורה 60.

10 INPUT "A =" ; A : INPUT "DIVIDED BY" ; B

60 ? : GOTO 10

LIST

5 REM-LONG DIVISION CRANKER OUTER

10 INPUT "A =" ; A : INPUT "DIVIDED BY" ; B

20 Q=INT(A/B)

30 R=A-B*Q

40 PRINT "QUOTIENT =" ; Q : PRINT "REMAINDER =" ; R

60 PRINT : GOTO 10

OK

RUN

A =? 97

DIVIDED BY? 5

QUOTIENT = 19

REMAINDER = 2

A =? 2435

DIVIDED BY? 14

QUOTIENT = 173

REMAINDER = 13

A =?

OK





קרא

הפונקציה השלישית שאנו רוצים ללמד אותך, היא הפונקציה $RND(X)$, השולפת מספרים מתוך כובע. הפונקציה RND מלאת תהפוכות. כשאתה מזין את המחשב בפונקציה RND הוא נותן לך מספר אקראי (הנראה כמו שבר עשרוני) בין אפס לאחד. לעולם לא 0, לעולם לא 1, תמיד ביניהם ידידי. אם המספר האקראי (השבר העשרוני) קטן מאוד, כלומר קרוב מאוד לאפס, ניתן להדפיס בסימון נקודה צפה. המספר, המשתנה או החישוב המצויים בסוגריים של פונקציות אלו מכונה לעתים הארגומנט של הפונקציה... יש שלושה סוגי ארגומנט שאפשר להכניס לסוגריים בפונקציה $RND(X)$, לשלוש מטרות שונות. עשה את התרגילים הבאים ותראה כיצד פועלת הפונקציה RND .



בצע

RND(1)

הארגומנט של RND הוא חיובי

NEW

OK

10 X=1

20 ? RND(X) : GOTO 20

RUN

.50438
.0267824
.388094
.569123
.720021
.209046
.599836

אל תצפה שמספרי
ה- RND יהיו זהים
לאלה. ככלות הכול,
הם אמורים להיות
מספרים אקראיים.

BREAK IN 20

OK

הדבר קרה מפני
שלחצנו על
CONTROL/C.

ועתה הרץ שנית את התכנית
והשווה בין מספרי ה- RND
בהרצה זו למספרי ה- RND
בהרצה הראשונה.

RUN

.744055
.460434
.433291
.27376
.701146
.590584
.457448

שים לב שמספרי ה- RND
מהרצה זו של
אותה תכנית שונים
לחלוטין.

BREAK IN 20

OK

RND(0)

הארגומנט של RND הוא אפס.

ועתה החלף שורה 10 בשורה

10 חדשה, ולאחר מספר RND

לדוגמא, עצור בעזרת:

CONTROL/C

10 X=0

RUN

.457448
.457448
.457448
.457448
.457448
.457448
.457448

רשום לפניך
את מספר RND
זה ואחר כך
התבונן במספר
 RND בהרצה
אחרונה. מעניין,
נכון?

BREAK IN 20

OK

ועתה הרץ שנית והפסק את
הביצוע באמצעות:
CONTROL/C והשווה להרצות
הקודמות.

RUN

.457448
.457448
.457448

BREAK IN 20

OK

RND(-1)

הארגומנט של RND הוא שלילי.

החלף שוב שורה 10, בצורה זו:

10 X=-.4

RUN

.803906
.803906
.803906

BREAK IN 20

OK

ושוב, עצור בעזרת:

CONTROL/C, ושוב RUN נוסף

והשווה אותו לכל השאר.

RUN

.803906
.803906

BREAK IN 20

OK

ועד ערך שלילי ל-x.

10 X=-1

RUN

7.65943E-06
7.65943E-06
7.65943E-06
7.65943E-06

מספר RND

"קטן (קרוב

לאפס) מודפס

בעזרת סימון

הנקודה הצפה.

BREAK IN 20

OK

והנה ערך שלילי נוסף.

10 X=-.3

RUN

.905996
.905996

המ..... מבוטל

נבדף המשך לקרוא

ותבין הכול.

BREAK IN 20

OK

מספרים



אקראיים (RND)

ווטסון יקירי, עכשיו נוכל להתחיל להסיק מן הניסויים האלה מהם החוקים שבעזרתם ניתן להשיג כל מה שתמצא מן הפונקציה RND.

ארגומנט חיובי.

לצורך קבלת מספר RND חדש ושונה, בכל פעם שהמחשב מבצע הוראה באמצעות פונקציה RND, חייב המספר שבסוגריים להיות גדול מאפס (כל מספר חיובי). תוכל להשתמש בכל מספר כבארגומנט, כדוגמת 1, שבו השתמשת בתכנית ה-RND לדוגמא, שהבאנו בעמוד הקודם.

אתה יכול להשתמש בשברים. עשרוניים או במספרים בעלי שברים עשרוניים, כדוגמת (3.52) RND, אך אין זה משנה לקבלת מספר RND חדש בכל פעם שמתבצעת הוראת RND.

סוג זה של פונקציה RND מתאימה לתכניות משחק, היכן שתמצא להכניס יסוד של סיכויים ומזל.

ארגומנט אפס.

השימוש ב-0 בתורת ארגומנט מביא לך שוב ושוב את אותו מספר RND. אך אם תתבונן בשימת לב בהרצות, תווכח שמספר RND שקיבלת אחר השימוש ב-0, זהה למספר RND האחרון שנוצר על-ידי RND. "נוצר" פירושו התקבל מביצוע הפונקציה על-ידי המחשב.

סוג זה של פונקציה RND יעיל לעתים לצורך הפעלת תכנית, שאינה פועלת, כאשר הינך מבקש למנוע את שינויו של מספר RND האחרון שנוצר על-ידי המחשב. כך תוכל להבחין בהשפעת משתנים או ערכים אחרים בתכנית; בעוד מספר RND נשאר ללא שינוי.



השתמש בגישה ישירה כדי לראות כיצד לוקח $RND(0)$ את מספר RND שנוצר אחרון.

$RND(0)$; $RND(-.15)$; $RND(0)$; $RND(-1)$; $RND(0)$; $RND(2)$; $RND(0)$; $RND(1)$; ?
 .603906 .7.65943E-06 .737525 .737525 .163989 .163989 .603906

OK

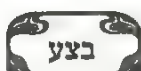
ועתה, אם רצונך בכך, נסה את הערכים שלך עם ארגומנטים שאינם אפס.



קרא

ארגומנט RND שלילי

השימוש במספר שלילי כבארגומנט RND נותן לך מספר RND חוזר בכל פעם שמתבצעת אותה הוראת RND. יחד עם זאת, לכל ארגומנט שלילי שונה, הינך מקבל מספר RND חוזר שונה. אתה יכול להשתמש בשברים עשרוניים גם כבארגומנטים שליליים, כגון $RND(-.3)$ או $RND(-9.32)$. בכל פעם שהינך מריץ את התכנית או מבצע הוראה עם אותו ארגומנט שלילי, אתה מקבל אותו מספר RND.



בצע

המחש כיצד עובד RND עם מספרים שליליים, בשימוש בגישה ישירה.

```
? RND(1); RND(-.4); RND(-.4); RND(-.4); RND(-17.5); RND(-990)
.905996 .803906 .803906 .803906 8.37562E-06 1.96788E-03
```

OK

נסה את המחשתך שלך, כשאתה בוחר בעצמך את הארגומנטים RND.



קרא

במקרים מסוימים, רצוי ויעיל להשתמש במספרי RND שבתכנית בכל פעם שמריצים אותה תכנית, כמו למשל בתכנית סימולציה. (בהמשך תשמע יותר אודות סימולציות.) למעשה, $RND(-.3)$ נותן למחשב מספר RND התחלתי להתחלת הרשימה, וה- $RND(1)$ בוחר וממשיך ליצר מספרי RND נוספים. נסה את תכניות החזרה שלנו.



בצע

NEW

OK

5 REM-RND NUMBER LIST REPEATER

10 X=RND(-.3)

20 ? RND(1)

30 GOTO 20

RUN

.905996

.270418

.504185

.885868

.22101

.457995

השתמש ב-CONTROL/C לעצירת ההרצה.

BREAK IN 20

OK

RUN

הרצה חדשה, רשימה זהה.

.905996

.270418

.504185

.885868

.22101

.457995

BREAK IN 20

OK

מספרים אקראיים שלמים



לא תמיד נוח לעסוק במספרים אקראיים שהינם שברים עשרוניים מזוהים בין 0 ל-1. בוא נאמר שאנו רוצים מספרים אקראיים בין 0 ל-9. עלידי שילוב מתוכם של הפונקציות RND ו-INT, אנו יכולים לקבל מ-BASIC מספרים אקראיים שלמים.

NEW

OK

5 REM-RND INTEGERS

10 X=RND(1) ← מספר RND נוצר ומוצב למשתנה X.

20 X=X*10 ← מספר RND מוכפל ב-10, המזיז את הנקודה העשרונית מקום אחד ימינה.

30 X=INT(X) ← INT קוצץ את השבר הלא רצוי, ויוצר בכך שלמים של RND.

40 ? X

50 GOTO 10

RUN

9	1	3	5	5	7	3	8	7	8	5	8	8	4	3	5	2	4	2	4	3	3	4	4
7	7	5	2	9	1	4	2	9	5	1	8	0	5	6	4	2	0	3	9	7	1	7	3
0	8	1	9	7	5	0	1	8	6	6	4	1	6	2	2	7							

BREAK IN 40

OK

ועתה הכנס גירסא מחודשת זו של התכנית, היא מדפיסה כותרות ואת השלבים ביצירת סדרת השלמים של RND.

NEW

OK

5 REM-RND INTEGERS EXPLAINED

10 ? " X=RND(1)", " X=X*10", " X=INT(X)"

20 X=RND(1) : ? X, איזו שורה של התכנית

30 X=X*10 : ? X, מדפיסה איזה טור ערכים?

40 X=INT(X) : ? X

50 GOTO 20

RUN

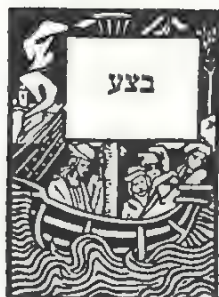
X=RND(1)	X=X*10	X=INT(X)
.743459	7.43459	7
.835007	8.35007	8
.0717177	.717177	0
.371099	3.71099	3
.271622	2.71622	2
.929463	9.29463	9
.232348		

BREAK IN 20

OK



R
A
N
D
O
M



ומה קורה כשאנו רוצים מספר RND מ-1 עד 10 במקום מספרים מ-0 עד 9? פשוט, עליך להוסיף 1 לערך של X. הדבר יכול להעשות בכל עת אחרי שנוצר מספר RND. שנה את התכנית RND INTEGERS EXPLAINED עלידי החלפת שורה 10, וכן את הכותרות. החלפת שורה 40 בהוספת פסיק לסוף השורה והכנסת שורה חדשה בין השורות 40 ו-50.

```
10 ? " X=RND(1)", " X=X*10", " X=INT(X)", " X=X+1"
40 X=INT(X) : ? X,
45 X=X+1 : ? X
```

אתה עורך LIST כדי לראות את התכנית בה הוכנסו השינויים. היא נראית

```

RUN
X=RND(1)      X=X*10      X=INT(X)      X=X+1
.951037      9.51037      9      10
.376529      3.76529      3      4
.220719      2.20719      2      3
.379046      3.79046      3      4
.818024      8.18024      8      9
.989366      9.89366      9      10
5.23684E-03  .0523684     0      1
.281303      2.81303      2      3

```

```

BREAK IN 45
OK

```

או מתחשק לך באמת לעשות זאת בשורה כדוגמת זו:

NEW

OK

```
10 ? INT(10*RND(1))+1 : GOTO 10
```

RUN

```

9 1 8 1 2 5 7 5 1 2 6 6 5 9 2 8 10 5 8 2 8 9 6
7 1 6 7 6 9 10 3 7 8 5 6 1 8 10

```

BREAK IN 10

OK

RUN

```

4 7 4 2 7 5 1 4 7 5 2 6 5 10 1 4 5 9 3 5 3 8 9
10 9 3 7 9 2 4 7 7 1 8 9 2 8

```

BREAK IN 10

OK

ומה בדבר מספרים אקראיים מ-1 ועד 100? פשוט מאוד!

```
10 ? INT(100*RND(1))+1 : GOTO 10
```

RUN

```

65 42 76 57 63 5 57 27 7 22 52 88 9 60 83 42 90 32
83 98 44 62 11 11 64 80 90 54 31 72 91 45 14 86 85 81
35 80 90 15 95 71 47 93 74 82 19 96 8 93 83 79 85

```

BREAK IN 10

OK

קדימה, נסה זאת עם מספרים מ-1 עד 1000. יש לך מרץ?
ומה בדבר מספרים אקראיים שבין 1 ל-6? ובין 50 ל-100?

וכעת, כשאתה מתקדם יפה כל כך בלימוד שפת ה-BASIC, נסה את התכנית הבאה, בה עליך להוכיח את כל הידע שצברת ב-BASIC במשחק שאתה וחבריך יכולים לשחק עם המחשב. כן חבר'ה, תכנית משחק אמיתית. לפני הדפסת התכנית, עבור עליה ונסה להבין אותה שורה אחרי שורה. אם תתבלבל, תוכל לחפש את ההסבר בעמוד הבא. שים לב שאנו משתמשים בהוראות REM, המסייעות לכל מי שמתעמק בתכנית כדי להבין את החלק הבא של התכנית. תזכור כשהתכנית מורצת, מדלג המחשב על הוראות ה-REM. גם אתה יכול לדלג עליהן כשהינך מדפיס תכנית, אם ברצונך לחסוך עבודה, זמן או מקום בזיכרונו של המחשב.

*
*

בצע (כשתבין מה עושה התכנית....)

NEW

OK

5 REM-NUMBER A GUESSING GAME

10 REM-ADAPTED FROM THE BOOK "WHAT TO DO AFTER YOU HIT RETURN"

200 REM-PRINT THE INSTRUCTIONS FOR THE GAME

210 ? "I WILL THINK OF A WHOLE NUMBER FROM 1 TO 100."

220 ? "TRY TO GUESS MY NUMBER. AFTER EACH GUESS, I WILL"

230 ? "TELL YOU IF YOU HAVE GUESSED MY NUMBER, OR IF YOUR"

240 ? "GUESS IS TOO SMALL OR TOO BIG."

300 REM-COMPUTER "THINKS" UP A NUMBER - CALL IT X

310 X=INT(100*RND(1))+1

320 ? : ? "OK, I HAVE A NUMBER. START GUESSING."

400 REM-PLAYER GUESSES, COMPUTER COMPARES AND DECIDES

410 ? : INPUT "WHAT IS YOUR GUESS"; G

420 IF G=X THEN ? "YOU GOT IT!!! LET'S PLAY AGAIN " : GOTO 310

430 IF G<X THEN ? "TOO SMALL. GUESS AGAIN." : GOTO 410

440 ? "TOO BIG. GUESS AGAIN." : GOTO 410

RUN

I WILL THINK OF A WHOLE NUMBER FROM 1 TO 100.

TRY TO GUESS MY NUMBER. AFTER EACH GUESS, I WILL

TELL YOU IF YOU HAVE GUESSED MY NUMBER, OR IF YOUR

GUESS IS TOO SMALL OR TOO BIG.

OK, I HAVE A NUMBER. START GUESSING.

WHAT IS YOUR GUESS? 50
TOO SMALL. GUESS AGAIN.

WHAT IS YOUR GUESS? 75
TOO BIG. GUESS AGAIN.

WHAT IS YOUR GUESS? 60
TOO BIG. GUESS AGAIN.

WHAT IS YOUR GUESS? 55
TOO BIG. GUESS AGAIN.

WHAT IS YOUR GUESS? 53
TOO BIG. GUESS AGAIN.

WHAT IS YOUR GUESS? 52
TOO BIG. GUESS AGAIN.

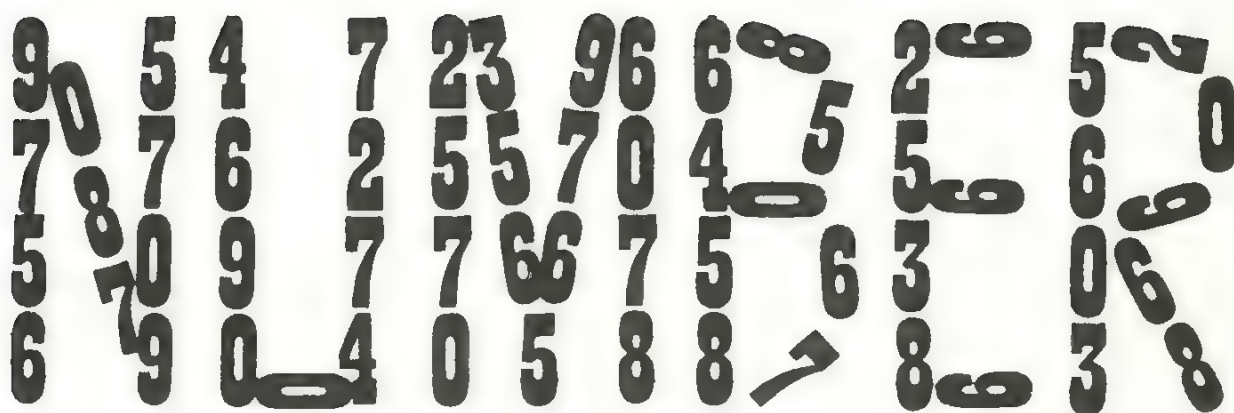
WHAT IS YOUR GUESS? 51
YOU GOT IT!!! LET'S PLAY AGAIN

OK, I HAVE A NUMBER. START GUESSING.

WHAT IS YOUR GUESS?

OK





המספר החספני

אל תשכח (ש-א) אתה יכול להשתמש ב-? כתחליף ל-PRINT כשאתה מדפיס הוראות (ב-PRINT שלאחריו אין כל הוראה, מותר שורה ריקה בהדפסה.

```
310 X=INT(100*RND(1))+1
```

שורה 310 בוחרת במספר אקראי שבין 1 ל-100 ומציבה אותו למשתנה X.

```
420 IF G=X THEN ? "YOU GOT IT!!! LET'S PLAY AGAIN " : GOTO 310
```

שורה 420 משווה בין מספר המחשב (ערכו של X) לניחוש המשחק (ערכו של G) ואם הם זהים (התנאי נכון) אומר לך המחשב "YOU GOT IT!", שפירושו "תפסת."

```
430 IF G<X THEN ? "TOO SMALL. GUESS AGAIN." : GOTO 410
```

שורה 430 מדפיסה TOO BIG אם ניחוש G גדול מ-X.

```
440 ? "TOO BIG. GUESS AGAIN." : GOTO 410
```

שורה 440 מבוצעת כך שתנאי ה-IF בשורות 420 ו-430 הן שקריות והמחשב מדלג לשורה 440.

שורה 440 מדפיסה TOO SMALL מפני שבמקרה שהניחוש אינו שווה או גדול ממספר המחשב X, הוא צריך להיות קטן יותר. המחשב אינו זקוק להוראת FI כדי להחליט על כך!

האם אתה מבין כיצד פועלת התכנית? אם כן, התחל להכניס ולהריץ אותה. מקווה שהכול יודפס כבר בניסיון הראשון. אם אינך מצליח להדפיס ולהריץ אותה, לחץ על הקליד LIST וחפש את הטעות שעשית.

מיוחד למומחים: פתח גירסא של המשחק לשני שחקנים המנחשים לסירוגין, עד שאחד מהם מגיע למספר הנכון.



מיוחד למומחים: פתח גירסא של המשחק לשני שחקנים המנחשים לסירוגין, עד שאחד מהם מגיע למספר הנכון.

פרק 6: בעיות

1. במקרה ששורה 20 בתכנית ביצוע הייתה `PRINT SQR(X)` וקיבלת `FC ERROR` בשורה 20, תוכל לשער מה הייתה הבעיה?

2. כיצד יחשב המחשב את ערך `INT` של המספרים שלהלן?

3.97 -3.96 4.1

-4.1 2 -2

3. רשום את ההוראה שתחשב ותדפיס 30 מספרים אקריים שונים שבין 1 ו-6. (כדוגמת 30 זריקות קוביה.)

4. רצה לכתוב תכנית? חזור לתכנית ה-`NUMBER`. הכנס את השינויים הדרושים כדי שהתכנית תמנה את מספר הניחושים. וכשהינך קולע, הדפס: "YOU GOT IT IN _____ GUESSES. LET'S PLAY AGAIN."
(בנקודה זו יחזיר המתכנת החכם את מונה הניחושים לאפס!)

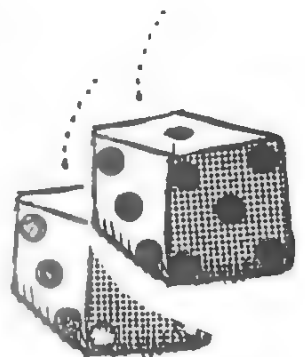
5. משחק קוביות פשוט ופופולארי בקרב אוהדי ההימורים בעולם הוא, זה שבו על השחקן לזרוק קוביה בת ש פאות כדי לקבל ניקוד, לחזור על כך שוב ושוב כדי לנסות להגיע שוב לאותו ניקוד. אם הוא הצליח (גם אתן, גברות יכולות לנסות) הוא מנצח. אם לא, ימשיך שחקן אחר לזרוק קוביה, עד שישגיג את הניקוד או את המספר 7. במקרה כזה אתה יוצא מהמשחק. ואז אתה שוב מתחיל בזריקת הקוביה להשגת ניקוד חדש, אם מספר הנקודות מסתכם ב-7 או ב-11 אתה המנצח! כתוב תכנית שתשחק גירסה פשוטה זו של המשחק אולי תרצה גם להוסיף לתכניתך חוקים המאפיינים את המשחק בשכונה שלך? כשתבחן את תכניתך, אתה עשוי להוסיף מספרים אקראיים, כדי לוודא שהכול כשורה.

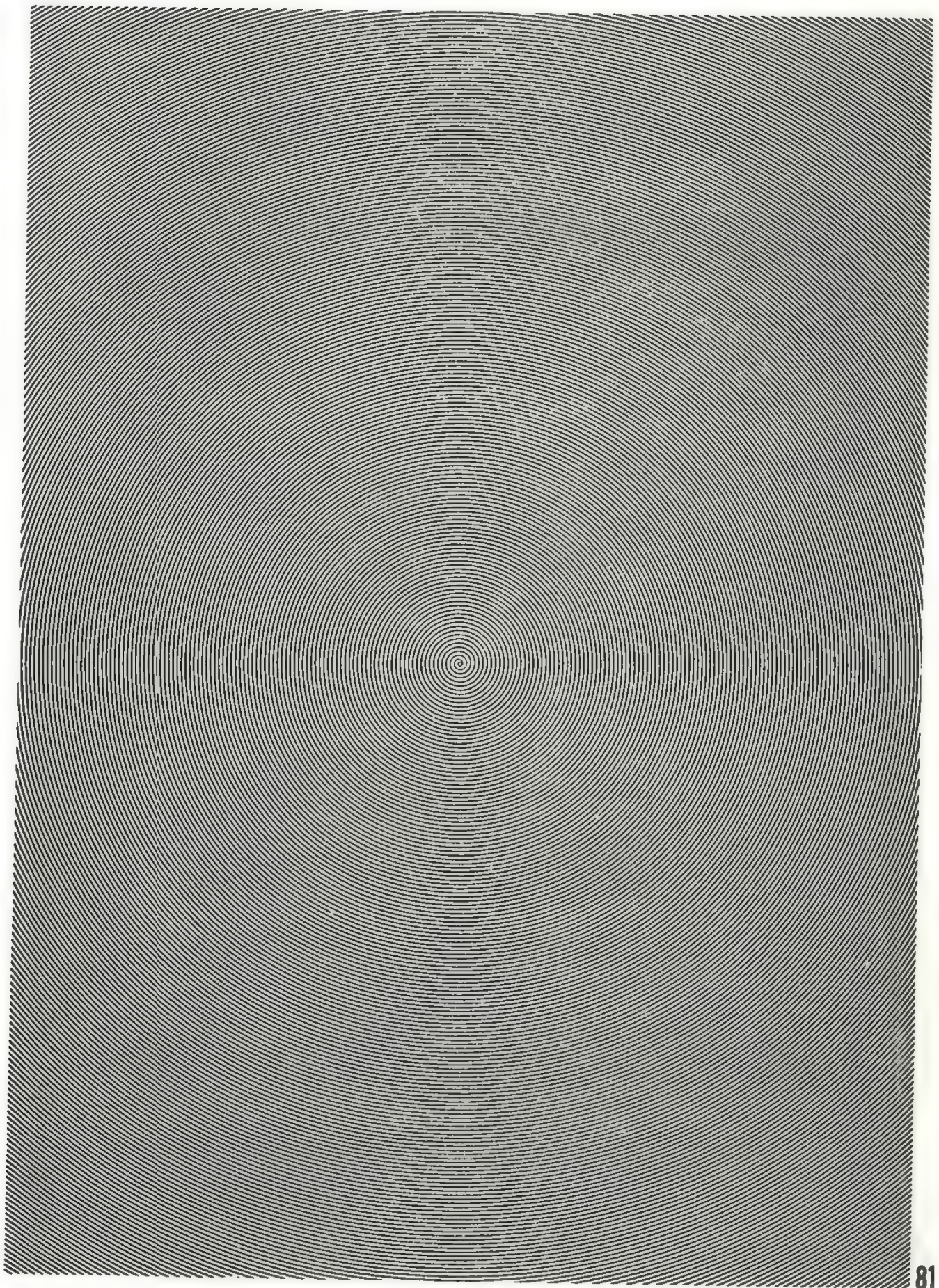
6. התחל במשחק ניחושים. להלן החוקים המודפסים על ידי המחשב. אינך צריך לכלול אותן בתכניתך.

"אחשוב על מספר שלם מ-1 עד 100. נסה לנחש מהו המספר. לאחר שתנסה מהו המספר, אדפיס כוכב אחד או יותר (*). ככל שאתה קרוב למספר שלי, כן אדפיס יותר כוכבים. כוכב אחד (*) פירושו שאתה רחוק מאוד מהמספר שלי. שבעה כוכבים (*****) מצביעים על כך שאתה קרוב מאוד מאוד למספר שלי!!"

הגיון: אם הניחוש סוטה מהמספר ביותר מאשר ב-64 מספרים - כוכב אחד; סטיה של 32-64 - שני כוכבים; סטיה של 16-31 - 3 כוכבים; סטיה של 8-15 - 4 כוכבים; סטיה של 4-7 - 5 כוכבים; סטיה של 2-3 - 6 כוכבים וסטיה של מספר 1 - שבעה כוכבים. יהיה עליך להשתמש בפונקציית הערך המוחלט, משהו חדש אך קל ביותר. רמז אחד ...

`IF ABS (X - Y) = 10 THEN 100`





לולאות אוטומטיות

קרא

והנה תכנית קטנה "הסופרת עד 7" ושלמעשה אומרת לך כמה לולאות היא עושה כשהתכנית מורצת.
עליך לשים לב לסופה של שורה 20, המכניסה לשימוש הוראת BASIC חדשה ונוחה. מה לדעתך היא מורה למחשב?

בצע

EW

```
OK
5 REM-AUTOMATIC LOOP COUNTER
10 F=1
20 IF F>7 THEN ? "NOW F ="; F : STOP
30 ? "F ="; F
40 F=F+1 : GOTO 20
```

RUN

```
F = 1
F = 2
F = 3
F = 4
F = 5
F = 6
F = 7
NOW F = 8
```

```
BREAK IN 20
OK
```

שורה זו נועדה לבדוק אם הלולאה חזרה על עצמה יותר מ-7 פעמים.

ערכו של F משתנה ב-1 בכל פעם שהוא עובר לולאה, וכך הוא מונה את מספר הלולאות.



קרא

האם אתה מבין כיצד פועלת התכנית דלעיל? אם כן, נסה סוג אחר של לולאה אוטומטית עלידי ביצוע התכניות שבעמוד הבא. התכניות האלו מעלות לולאות FOR-NEXT. הן מכונות לולאות FOR-NEXT, מפני שהן משתמשות בהוראות FOR ו-NEXT. נראה לכם שתי דרכים שונות לכתיבת אותה תכנית. הגירסא הראשונה משתמשת בשורת הוראה אחת, ואילו הגירסא השנייה - בריבוי הוראות, להכנסת הוראות FOR-NEXT לתוך שורה אחת.

אנו מציגים FOR-NEXT



NEW

OK

5 REM-USING THE FOR-NEXT LOOP CONTROL VARIABLE TO COUNT OFF LOOPS

10 FOR F=1 TO 7 ← FOR זוהי הוראת

20 ? "F ="; F

30 NEXT F ← NEXT זוהי הוראת

40 ? "NOW F ="; F

RUN

F = 1

F = 2

F = 3

F = 4

F = 5

F = 6

F = 7

NOW F = 8

OK

גירסא ראשונה

ועתה בצע את התכנית הבאה והשווה לזו שביצעת זה עתה.
שורות 10, 20 ו-30 בגירסא הראשונה מופיעות כולן בשורה מרובת
הוראות אחת. שורה 10 בגירסא השניה.

NEW

OK

10 FOR F=1 TO 7 : ? "F ="; F : NEXT F

20 ? "NOW F ="; F

RUN

F = 1

F = 2

F = 3

F = 4

F = 5

F = 6

F = 7

NOW F = 8

OK

גירסא שניה



הלולאה FOR~NEXT

מוסברת



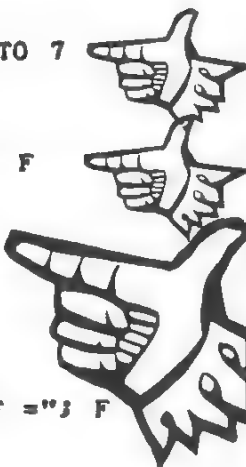
לולאת FOR-NEXT פועלת כך. F הוא משתנה הפיקוח ללולאה FOR-NEXT. לך בעקבות החצים.

10 FOR F=1 TO 7

20 ? "F =" ; F

30 NEXT F

40 ? "NOW F =" ; F



יש להציב את הערך 1 למשתנה F. ואז עליך לעבור להוראה הבאה בתכנית. 1 (אחד) הוא ערכו ההתחלתי של F ו-7 הוא הגבול העליון של F, והוא גם מספר הפעמים שהלולאה חוזרת על עצמה. הדפס את ערכו של F. F=1 בפעם הראשונה בתוך הלולאה. ההוראה NEXT מראה למחשב היכן מסתיימת הלולאה, ושולחת אותו חזרה אל ההוראה FOR (שורה 10). ערכו של F גדל ב-1 ($F=F+1$) והמחשב מתחיל לחזור אחורה, דרך הלולאה. כש-F עולה על 7 (הגבול העליון של השמתנה F), מדלגת התכנית להוראה הבאה בתכנית אחרי הלולאה FOR-NEXT. פירוש הדבר שהיא תדלג על שורות 20 ו-30 ותבצע את שורה 40 (ההוראה שאחרי ההוראה NEXT).

זוהי הצורה הכללית של הלולאה FOR-NEXT.



FOR C=A TO B ההוראה - FOR



קובעת את ערכם ההתחלתי והסופי של C, כלומר היא קובעת מהו ערכו הראשוני של C ומאיזה גבולות הוא אינו יכול לחרוג.

גוף הלולאה, הוראות המבוצעות שוב ושוב עד שהלולאה מסתיימת. יכול להיות מורכב מהוראה אחת או יותר.

NEXT C ההוראה - NEXT



סוגרת את הלולאה והמחשב חוזר להוראה FOR: אם $C < B$, הוא חוזר ועובר בלולאה.



ההוראה FOR חייבת להיות מלווה תמיד בהוראת NEXT תואמת, כדי להורות למחשב היכן מסתיימת הלולאה ולהחזירו אל ההוראה FOR, כדי להתחיל שנית בלולאה. יחד עם זאת, ניתן להשמיט את המשתנה אחרי NEXT, אם תכניתך פשוטה והוצאת המשתנה אחרי NEXT אינה מבלבלת אותך ואת המחשב שלך.

תירגול, תירגול ועוד תירגול



להלן מספר תכניות שבעזרתן תוכל לתרגל לולאות FOR-NEXT.

NEW

OK

10 A=5 : B=10

20 FOR C=A TO B : ? "C ="; C : NEXT C

RUN

C = 5

C = 6

C = 7

C = 8

C = 9

C = 10

OK

החלף שורה 10 בשורה שבהמשך.

10 INPUT "A ="; A : INPUT "AND B ="; B

והוסף שורה 30.

30 ? : GOTO 10

LIST

10 INPUT "A ="; A : INPUT "AND B ="; B

20 FOR C=A TO B : PRINT "C ="; C : NEXT C

30 PRINT : GOTO 10

OK

RUN

A =? 4

AND B =? 8

C = 4

C = 5

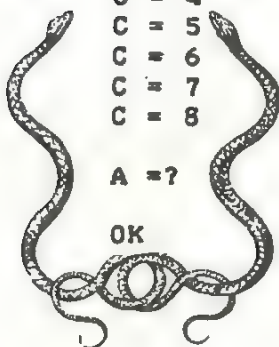
C = 6

C = 7

C = 8

A =?

OK



נסה ערכים אלה, אך אל תעצור. מבין את הרמז?

A=0, B=6

A=-9, B=2

● A=3.3, B=7.5

A=6, B=3

וכו', וכו', וכו'.....

TILT!

צעד אחר צעד

אתה רואה שערכו של משתנה הלולאה עולה אוטומאטית ב-1, בכל פעם שהוא עובר בלולאה. אך גם ה-BASIC מאפשר לך לשלוט בשיעור גידולו. יש כאן לולאה FOR-NEXT חדשה ומתוחכמת יותר, עם שלושה ארגומנטים או פרמטרים, למשתנה הלולאה FOR-NEXT.



`20 FOR C=A TO B STEP S` של העליה של S הוא שיעור משתנה C בכל פעם שהוא עובר בלולאה. ערכו הסופי של C. ערכו ההתחלתי משתנה הלולאה FOR-NEXT של C.



ערכו של S ב-STEP S יכול להיות מספר שלם או שבר.

צעד אחורה

תן דעתך לכך, שהערך של משתנה הפיקוח FOR-NEXT עשוי לקטון בכל פעם שהוא עובר בלולאה. הדבר מבוצע על-ידי השימוש בערך שלילי ל-STEP. אך התכנית לא תורץ בלולאות יותר מפעם אחת, במקרה שערכו של A קטן מערכו של B. בפעם הראשונה שהמחשב מגיע ל-NEXT ומשווה בין ערך C לערך B ומגלה ש-C קטן יותר, הוא מסיים את הלולאה וממשיך לאותו חלק התכנית שאחרי הלולאה FOR-NEXT.



תירגול נוסף מוצע לך

86 בעמוד הבא

נסה את הערכים הבאים וכל מה שעוד תרצה לנסות.

A=1, B=10, STEP=2
A=1, B=6, STEP=.7
A=6, B=1, STEP=-1.5
וכו', וכו', וכו'.....



NEW

OK

5 REM-FOR STATEMENT WITH STEP

10 INPUT "A="; A : INPUT "B="; B : INPUT "STEP="; S

20 FOR C=A TO B STEP S : ? "C="; C : NEXT C

30 ? "OUT OF THE LOOP BECAUSE C="; C : GOTO 10

RUN

A=? 1

B=? 10

STEP? 2

C = 1

C = 3

C = 5

C = 7

C = 9

OUT OF THE LOOP BECAUSE C = 11

A=? 1

B=? 6

STEP? .7

C = 1

C = 1.7

C = 2.4

C = 3.1

C = 3.8

C = 4.5

C = 5.2

C = 5.9

OUT OF THE LOOP BECAUSE C = 6.6

A=? 6

B=? 1

STEP? -1.5

C = 6

C = 4.5

C = 3

C = 1.5

OUT OF THE LOOP BECAUSE C = 0

A=?

OK



עצרנו כאן (תוך שימוש ב-RETURN כדי לצאת מלולאת INPUT), אך
אנו באמת ובתמים מקווים שאתה לא עצרת!!!

צפרדעים?? ציחקוקים? תהיה רציני!!!

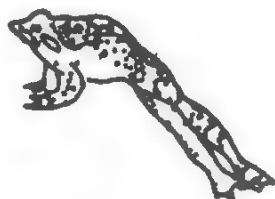


נסה עוד כמה לולאות FOR-NEXT. נסה לשנות את גוף הלולאה, כדי שתעשה דברים נוספים מלבד להראות לך בכל פעם את ערכו של C. נסה!

נסה זאת באמצעות תכנית זו, שנושאה צפרדעים.

NEW

```
OK
5 REM-TALKING FROG PROGRAM
10 FOR K=1 TO 5
20 ? "GRIBBIT"
30 NEXT K
RUN
GRIBBIT
GRIBBIT
GRIBBIT
GRIBBIT
GRIBBIT
```

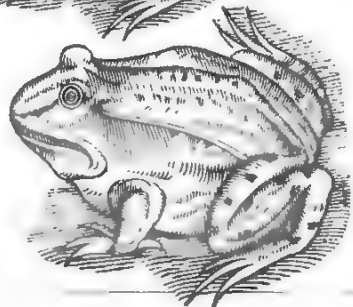
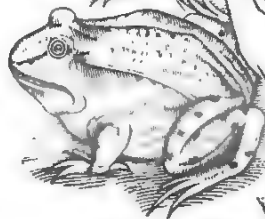
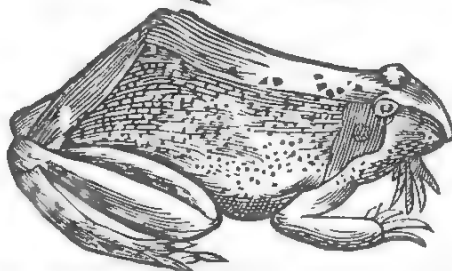
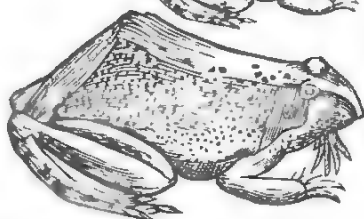
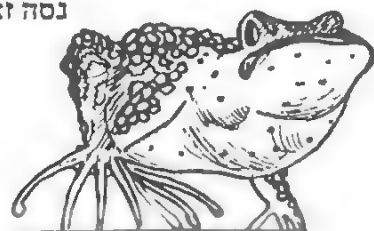


OK

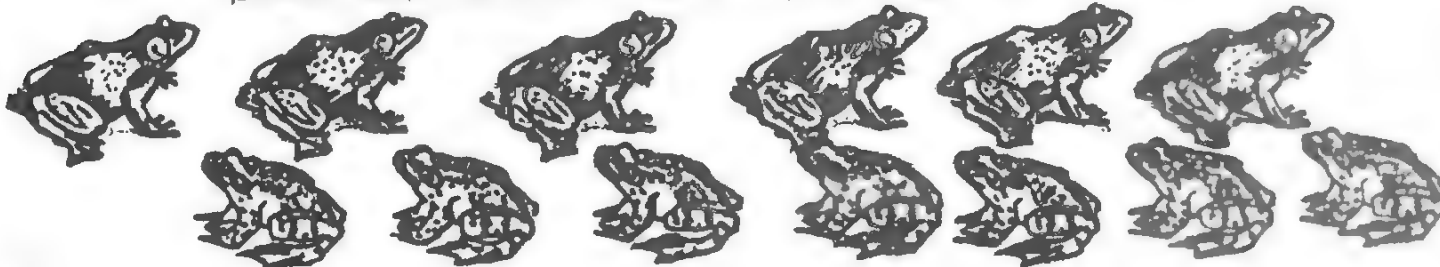
רעתה תכנית סימולציה.
בריכה המלאה בצפרדעים (מי אוהב
שוקי צפרדעים?)

NEW

```
OK
5 REM-TALKING FROG POND
10 FOR K=1 TO 200
20 ? "GRIBBIT ";
30 NEXT K
RUN
GRIBBIT GRIBBIT GRIBBIT
BREAK IN 20
OK
```



הנחה מיוחדת לשונאי צפרדעים: אתם יכולים להשתמש ב-C/CONTROL כדי לעצור את התקדמותה של הלולאה FOR-NEXT, אם רצונכם בכך.



בגישה ישירה



גרה את דימוינו של המחשב שלך בהוראה זו, בגישה ישירה, כדי להצחיק את המחשב.

[illegible]

ועתה השתמש בגישה ישירה כדי להדפיס טבלת ריבועים ושורשים ריבועיים של מספרים מ-1 עד 16.

X	X ²	SQR(X)
1	1	1
2	4	1.41421
3	9	1.73205
4	16	2
5	25	2.23607
6	36	2.44949
7	49	2.64575
8	64	2.82843
9	81	3
10	100	3.16228
11	121	3.31663
12	144	3.4641
13	169	3.60555
14	196	3.74166
15	225	3.87298
16	256	4

נסה ליישם את רעיונותיך שלך בלולאות FOR-NEXT. עשה זאת עכשיו!



המלה האחרונה

בפרמטרים עבור משתנה
הפיקוח של הלולאה
FOR-NEXT

קרא

בצע

A=16 : Y=2.185 : FOR X=SQR(A) TO A/2 STEP INT(Y) : ? X : NEXT X

4

6

8

OK

הפרמטרים עבור משתנה הפיקוח עשויים להיות ביטויים הדורשים חישוב. המחשב מבצע את החישוב בפעם הראשונה שהוא נתקל בהוראת ה-FOR במהלכה של הרצת תכנית, ממשיך לפני שהוא מתחיל לבצע את גוף הלולאה FOR-NEXT הארגומנט FOR אינו משתנה, דהיינו הוא אינו מחושב מחדש כל אימת שהמחשב עובר בלולאה במהלך אותה הרצה.

לולאות FOR-NEXT נוחות לכל פעולה חוזרת. בוא נשתמש בלולאה FOR-NEXT כדי להדפיס טבלת נתונים. גוף הלולאה מורכב מנוסחה לחישוב ריבית על כסף, כדוגמת חשבון חיסכון בבנק. הנוסחה נשארת ללא שינוי, בעוד המשתנים שלה עשויים להשתנות בכל מעבר בלולאה. אנו משתמשים בפונקציה STEP כדי לבדוק אם החיסכון שלנו צובר ריבית בשיעורי ריבית שונים (משתנה 1 בתכנית שבהמשך). אנו רוצים לדעת כיצד גדל החיסכון שלנו אם שיעור הריבית מוגדל ב-1/4% על שיעור ריבית קיים, הנע בין 5% - 8%.



NEW

OK

```
5 REM-SIMPLE INTEREST COMPOUNDED YEARLY, 5 TO 8% IN 1/4% STEPS
10 REM-PRINT HEADINGS AND TABLE OF RETURN ON $500 IN 5 YEARS
100 PRINT "INTEREST", "PRINCIPAL + INTEREST"
110 PRINT "RATE", " (FIVE YEARS)"
120 FOR I=5 TO 8 STEP .25
130 R=500*(1+I/100)^5
140 PRINT I, R
150 NEXT I
```

OK

RUN

INTEREST
RATE

PRINCIPAL + INTEREST
(FIVE YEARS)

5	638.141
5.25	645.774
5.5	653.479
5.75	661.259
6	669.113
6.25	677.041
6.5	685.044
6.75	693.121
7	701.276
7.25	709.507
7.5	717.815
7.75	726.199
8	734.664

שים לב לכותרות הטבלה המרווחת, העושות שימוש בשתי הוראות PRINT (שורה 100 ושורה 110).

בצע זאת בכוחות עצמך: השתמש בידע הנרחב שרכשת ב-BASIC כדי לעגל את הערכים המודפסים, עד הסנט הקרוב ביותר.



לולאות בתוך לולאות



אתה יכול להשתמש בלולאה FOR ... NEXT אחרת. בשפת המחשב אומרים שהלולאה מוכנסת בתוך לולאה אחרת. למרות זאת, אינך יכול "לחבר" שתי לולאות, כלומר להציב את ה-NEXT עבור ה-FOR השני. נקל להבין זאת כשמתבוננים בכך.

נכון

NEW

OK

5 REM-NESTED FOR-NEXT LOOPS

10 FOR A=1 TO 3

20 FOR B=1 TO 5

30 ? "NESTED LOOPS"

40 NEXT B

50 NEXT A

RUN

NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS
NESTED LOOPS

OK

לולאה זו
מוכנסת בתוך
לולאה זו.



לא נכון

5 REM-CROSSED FOR-NEXT LOOPS

10 FOR A=1 TO 3

20 FOR B=1 TO 5

30 PRINT "CROSSED LOOPS"

40 NEXT A

50 NEXT B

RUN

CROSSED LOOPS

CROSSED LOOPS

CROSSED LOOPS

לולאות אלו חופפות,
ולכן אינן מוכנסות,
האחת בתוך השניה.

?NF ERROR IN 50

OK .FOR NF ERROR פירושו NEXT בלי

המחשב הגיע להוראה NEXT, אך הואיל והלולאות היו מוצלבות, הוא לא הצליח למצוא את הוראת FOR עם אותו משתנה פיקוח.

שים לב שהמחשב אכן ביצע את הלולאה FOR-NEXT הראשונה עם משתנה הפיקוח A. אך לאחר מכן פעל המחשב על פי ההוראות המפעילות את הלולאות - FOR-NEXT. לאחר הפעם השלישית בתוך הלולאה, A = 4, עבר המחשב להוראה הבאה, אחרי 40 NEXT A. פירוש הדבר שהוא הגיע ל- 50 NEXT B ישירות מלולאה A. המחשב סבר שהוא מצא הוראת NEXT (NEXT B) בלי הוראת FOR, עם משתנה פיקוח זהה. על כן הודיע לנו ש- NF ERROR IN 50. אם אדם פיקח כמון מתבלבל, שער בנפשך מה קורה למכונה טיפשה כמו המחשב.

לולאות מוכנסות



להלן שתי הדגמות נוספות של המנגנון המפעיל את לולאות FOR-NEXT. השתדל לבחון בשימת לב איזו לולאה אחראית לפלט, כשהינך מריץ את התכנית. תן דעתך שבכל מסע של הלולאה החיצונית, עוברת הלולאה הפנימית את המחזור השלם של הלולאות.

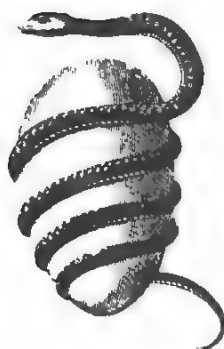


NEW

```
OK
5 REM-NESTED LOOPS REVISITED
10 FOR A=1 TO 3
20 PRINT "NESTED"
30 FOR B=1 TO 4
40 PRINT "          LOOPS"
50 NEXT B
60 NEXT A
RUN
```



FOR A=1 TO 3



```
NESTED
    LOOPS
    LOOPS
    LOOPS
    LOOPS } FOR B=1 TO 4

NESTED
    LOOPS
    LOOPS
    LOOPS
    LOOPS } FOR B=1 TO 4

NESTED
    LOOPS
    LOOPS
    LOOPS
    LOOPS } FOR B=1 TO 4
```

OK



NEW

```
OK
5 REM-CONTROL VARIABLE VALUES
10 FOR A=1 TO 3
20 FOR B=1 TO 5
30 PRINT "A ="; A; "B ="; B;
40 NEXT B
50 NEXT A
RUN
```

רואה? התבונן בשורת הפלט הראשונה. ערכו של משתנה הפיקוח A נשאר 1 עד שהלולאה B מגיעה לגבול (B=1 TO 5). ועתה הבט בשורת הפלט השנייה. A גדל ב-1 כך שעתה A=2, בעוד B מגיע שנית לגבולו. וכך הלאה גם עבור שורת הפלט הבאה.

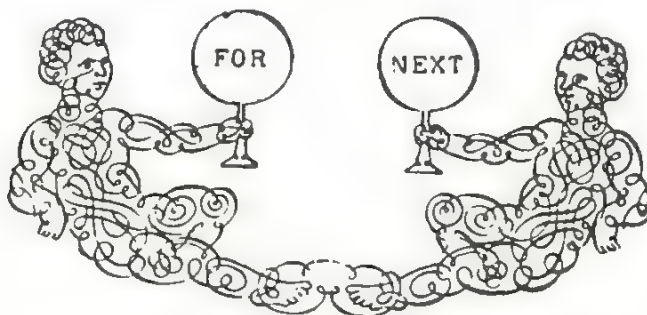


A = 1 B = 1	A = 1 B = 2	A = 1 B = 3	A = 1 B = 4	A = 1 B = 5
A = 2 B = 1	A = 2 B = 2	A = 2 B = 3	A = 2 B = 4	A = 2 B = 5
A = 3 B = 1	A = 3 B = 2	A = 3 B = 3	A = 3 B = 4	A = 3 B = 5

OK

שים לב לכך שכאשר אתה משתמש ביותר מלולאת FOR-NEXT אחת (ובעיקר בלולאות מוכנסות), כדאי לציין אחרי NEXT איזה משתנה FOR יוגדל עלידי הוראת ה-NEXT, כדוגמת NEXT A עבור A- משתנה ה-FOR הראשון ו-NEXT B- עבור B- משתנה ה-FOR השני.

עדיין רוצה להתנסות בהרפתקאות? נסה תכנית המכילה NEXT בלבד, דהיינו, בלי המשתנה אחרי NEXT.



ההוראה FOR-NEXT

מבצעת לולאה מבוקרת בין ההוראות FOR ו-NEXT, במשך מספר פעמים שצויין מראש.

TEP expression ביטוי עד ביטוי STEP (משתנה) = FOR (משתנה) מס' שורה
(משתנה) NEXT (מס' שורה)
1 הוא ערכו הראשוני של X, 30 הוא ערכו הסופי של X; אין מציינים STEP, על כן משתמע שהוא אחד (1). התכנית תדפיס את הערכים 1-30 ואז תמשיך להוראה הבאה- (END)

10 FOR 2 = N TO M ערכיהם של M ו-N הוגדרו בוודאי קודם לכן.

10 FOR Z = 1 TO 2 STEP .1 תדפיס את ערך 1-2 בהוספה של 1
20 PRINT Z, 1.1, 1.2, 1.3, 1.4, 1.5,.....20
30 NEXT Z

10 FOR X = 30 TO N STEP -1 תמנה כלפי מטה (הוספות שליליות) מ-30 עד N.

פרק 7 - בעיות

1. מה קורה בתכנית זו?

```
10 FOR X = 17 TO 5
20 PRINT X,
30 NEXT X
```
2. מה לקוי בתכנית זו, שמטרתה הוא להדפיס טבלה של 20 מספרים אקראיים?

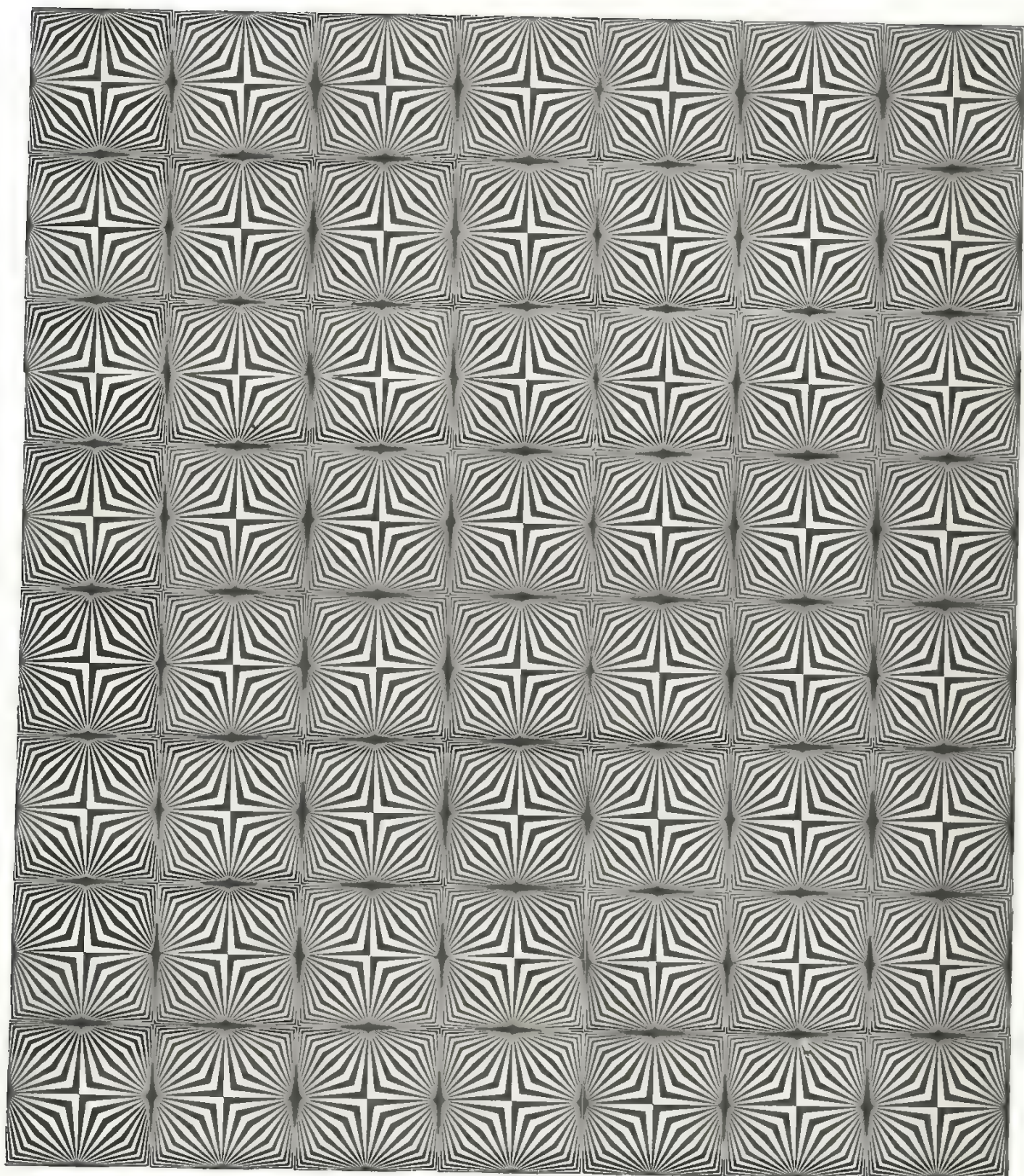
```
10 FOR X = 1 TO 20
20 PRINT "NO.", "RANDOM NO."
30 PRINT X, RND(0)
40 NEXT
```
3. תקן את תכנית ניחוש המספרים שבפרק 6, והגבל את המשתתף ל-8 ניחושים בלבד, וזאת באמצעות לולאת FOR-NEXT. אם המשתתף אינו מצליח לנחש נכונה אחרי 8 הניחושים, הדפס מסר מתאים והתחל מחדש את המשחק עם מספר חדש.
4. כתוב תכנית המשתמשת בלולאת FOR-NEXT, שתדפיס שורה מאוזנת של *, שתשתרע לרוחב השורה כולה.

5. תכניות החיסכון וההלוואות הן תחרותיות מאוד ונותנות לך ברירה בין גובה הריבית שתוצה להרוויח, לבין משך הזמן שעליך להשאיר בבנק את סכום הכסף הנ"ל כדי להגיע לסכום המיוחל. כתב תכנית שתשתמש ב-לולאות FOR-NEXT מוכנסות ושתדפיס טבלה, כמו בהמשך. הנח שיש ברשותך סכום של 10,000 ושהריבית מתווספת אחת לשנה (אם אתה יודע כיצד לחשב ריבית, אנא עשה זאת!)

שנים	5%	5.50%	6.00%	6.50%
5				
10				
15				
20				
25				

6. אתה מעסיק 20 עובדים. המידע המופיע בגליון התשלומים תורגם ל-20 הוראות DATA כדלקמן: DATA 900 שם (NS), מספר שעות העבודה, השכר לשעה.

כתוב תכנית שתמחשב את השכר השבועי ברוטו לכל עובד, והדפס את התוצאות בצורת דין וחשבון פשוט. אתה משלם לעוזר שלך 150% על כל השעות שמעבר למכסת 40 שעות העבודה השבועיות.



מסעך פונקציות #2

שפת ה-BASIC מעמידה לרשותך מגוון פונקציות לטיפול במחרוזות. בפרק זה נשתעשע במחרוזות ומשחקי מלים. כמו כן נציג בפניכם גם פונקציות מיוחדות אחרות.

LEN(<sup>מחרוזת
או
משתנה
מחרוזת</sup>)



LEN (XS) היא אותה פונקציה של BASIC המביאה את המחשב לידי כך שייגיד לך מה אורכה של מחרוזת. (זכור שהרווחים וסימני הפיסוק מהווים אף הם תווים בתוך מחרוזת, זאת מנקודת ראותו של המחשב.)

NEW

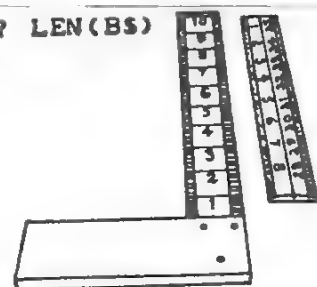


```
OK
10 INPUT "YOUR NAME";N$
20 ? N$; " HAS"; LEN(N$); "CHARACTERS."
RUN
YOUR NAME? JERALD R. BROWN
JERALD R. BROWN HAS 15 CHARACTERS.
```

OK

NEW

```
OK
10 BS="INSTANT BASIC IS EASY" : ? LEN(BS)
RUN
21
```



WORD

משחקי

ועכשיו, הבה נכנים שלוש פונקציות חדשות לפעולה, כדי שנוכל אחר כך להסבירן בפרוטרוט. את הרמז הראשון על מהותן של הפונקציות האלו תקבל מהתוועדות לשמותיהן: LEFT\$, MID\$ ו-RIGHT\$. שים לב כיצד אנו משתמשים בערך של LEN(IS) כדי לקבוע את הסייג למשתנה הפיקוח של הלולאה FOR-NEXT.



LEFT\$()

NEW

```
OK
10 IS="INSTANT BASIC"
20 FOR K=1 TO LEN(IS)
30 ? LEFT$(IS,K)
40 NEXT K
RUN
I
IN
INS
INST
INSTA
INSTAN
INSTANT
INSTANT
INSTANT B
INSTANT BA
INSTANT BAS
INSTANT BASI
INSTANT BASIC
```

OK

שים לב לשימוש
במשתנה הפיקוח K,
הבא כדי לפרט כמה
תווים מודפסים על ידי
הפונקציה LEFT\$.



RIGHT\$()

ועתה החלף שורה 30
והרץ שנית את התכנית.
(הדפס LIST, אם רצונך
בכך.)

```
30 ? RIGHT$(IS,K)
```

```
RUN
C
IC
SIC
ASIC
BASIC
BASIC
T BASIC
NT BASIC
ANT BASIC
TANT BASIC
STANT BASIC
NSTANT BASIC
INSTANT BASIC
```

OK

חלים

CATTEC

MID\$()

פונקציה נוספת לטיפול במחרוזות מופיעה בשורה 30. הדפס LIST אם הינך רוצה לראות את התכנית בשלמותה, לפני הרצתה.



```
30 ? MID$(IS,K,1), MID$(IS,K,3)
```

```
RUN
```

```
I  
N  
S  
T  
A  
N  
T
```

```
B  
A  
S  
I  
C
```

```
OK
```

```
INS  
NST  
STA  
TAN  
ANT  
NT  
T B  
BA  
BAS  
ASI  
SIC  
IC  
C
```

הקפד על ההבדל שבין שתי הפונקציות MID\$. מה תוכל לספר לי על הארגומנט השלישי שבתוך הסוגריים של MID\$?



האם הצלחת להבין מהן ההוראות שנותנות הפונקציות LEFT\$, LEN ו-RIGHT\$ למחשב?

נותנת לך את מספר התווים (ונגזרת מהמלה LENGTH שפירושה - אורך) במחרוזת, המוצבים למשתנה המחרוזת שהינך מכניס לסוגריים. LEN ()

נותנת לך את כל התווים במחרוזת, המזוהים על ידי משתנה המחרוזת, כשהיא מתחילה במספר התווים שאחרי משתנה המחרוזת בתוך הסוגריים. לדוגמא, LEFT\$(X\$,4) מורה להחזיר את אותו חלק במחרוזת שהוצב ל-X\$, כולל התו הרביעי משמאל (LEFT באנגלית) וכל אותם תווים, שמאלה בתוך המחרוזת. LEFT\$ ()

העזרת בגישה ישירה כדי להמחיש את דרך פעולתה של LEFT\$. ראשית, הצב מחרוזת של מספרים למשתנה המחרוזת X\$ ודאג לכך שהמחשב יחזיר רק את 4 התווים השמאליים של המחרוזת. לאחר מכן הצב את המחרוזת - "COMPUTERS" למשתנה המחרוזת Y\$ ודאג לכך שהמחשב יחזיר לך את 4 התווים השמאליים של מחרוזת זו.



```
X$="123456789" : ? LEFT$(X$,4)  
1234
```

```
OK
```

```
Y$="COMPUTERS" : ? LEFT$(Y$,4)  
COMP
```

```
OK
```





נותנת לך את כל התווים במחרוזת, כשהיא מתחילה בתו בעל ציון המקום המזוהה, וממשיכה לקצה הימני של המחרוזת. $RIGHT(X\$, 4)$ מורה לשוב אל חלקה של המחרוזת שהוצב ל- $X\$$, ולהתחיל בתו הרביעי מקצה הימני של המחרוזת ולהמשיך בכל התווים שלימינה של המחרוזת.

$RIGHTS()$

כשהמחרוזת שהוצבו ל- $X\$$ ו- $Y\$$ עדיין מצויות בזכרוננו של המחשב, תוכל להכניס הוראות גישה ישירה כדי להמחיש את דרך פעולת ה- $RIGHTS$.

? $RIGHTS(X\$, 4)$
6789

OK

? $RIGHTS(Y\$, 4)$
TERS

OK



מחזירה אותו חלק מחרוזת שזוהה עלידי המספרים שאחרי מחרוזת המשתנה, בתוך הסוגריים. המספר הראשון מורה היכן להתחיל והשני מורה כמה תווים יש להדפיס. אם אין מספר שני בארגומנט $MIDS$, מתחיל המחשב בתו שצויין ומדפיס תו זה ואת המשך המחרוזת.

$MIDS()$

דוגמא: $(X\$, 4)$ מורה להחזיר אותו חלק המחרוזת המתחיל בתור הרביעי, והכולל את המשכה של המחרוזת. $MIDS(X\$, 4, 3)$ מורה להחזיר את חלקה של המחרוזת המתחיל בתו הרביעי, אך רק תו זה ושני התווים הבאים אחריו (בסך הכול מוחזרים 3 תווים).

כשמחרוזת האותיות $X\$$ ומחרוזת האותיות $Y\$$ עדיין שמורות בזכרוננו של המחשב, השתמש בגישה ישירה לצורך המחשת פעולתה הכפולה של הפונקציה $MIDS$ על שתי המחרוזות.

? $MIDS(X\$, 4)$, $MIDS(X\$, 4, 3)$
456789 456

OK

? $MIDS(Y\$, 4)$, $MIDS(Y\$, 4, 3)$
PUTERS PUT

OK



בלבל אותם

תוכל לבלבל את התווים במחרוזות ולסדרם מחדש באמצעות השימוש בפונקציות אלו, כשאתה מחבר אותם על-ידי הסימן + (פלוס).



המצא גידוף משלך:

```
10 MS="HELLO TODAY, GOODBYE"
20 XS=MIDS(MS,14,2)+MIDS(MS,6,1)+MIDS(MS,7,2)+MIDS(MS,13,1)+LEFTS(MS,4)
30 PRINT XS
```

OK
RUN
CENSORED

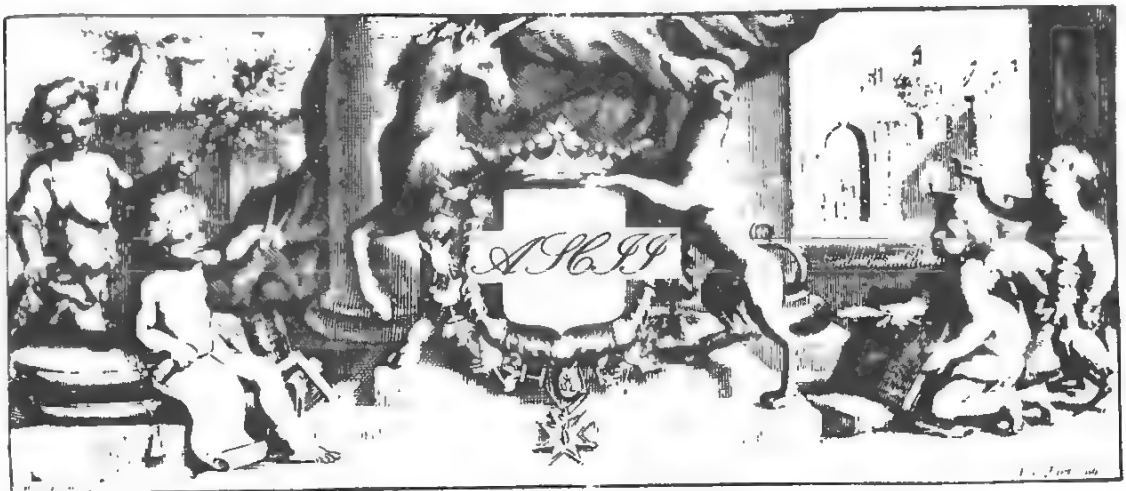
OK
NEW

השתמשנו במשתנה הפיקוח K בפונקציה MIDS, כדי לשלוף כל אות בתורה, ולהציבה ל-AS.

```
OK
10 XS="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
20 FOR K=1 TO LEN(XS) : AS=MIDS(XS,K,1) : ? AS; " = "; ASC(AS) : NEXT K
RUN
```

לכל תו שעל המקלדת, יש שווה ערך מספרי במחשב, המכונה מספר ASCII שלו. מהו ה-ASCII? שם זה הוא ראשי תיבות של: American Standard Code for Information Interchange (חשבתי שתהיה מעוניין לדעת זאת). הפונקציה ASC נותנת את מספר ה-ASCII עבור תווי המחרוזת.

A = 65
B = 66
C = 67
D = 68
E = 69
F = 70
G = 71
H = 72
I = 73
J = 74
K = 75
L = 76
M = 77
N = 78
O = 79
P = 80
Q = 81
R = 82
S = 83
T = 84
U = 85
V = 86
W = 87
X = 88
Y = 89
Z = 90



OK

שתיירות סודות



אפשר לומר ש-ASC(X) נותנת את ערכו המספרי ASCII לכל מספר. והדרך הפוכה היא להשתמש ב-CHR\$(X) כדי לשנות ערך מספרי לתו שווה הערך ASCII שלו. אתה יכול להשתמש בפונקציה זו כדי לשנות מסרים המקודדים במספרי ASCII לתווים שווי הערך שלהם.

5 REM-SECRET CODE DECODER

10 READ A

20 PRINT CHR\$(A); : GOTO 10

30 DATA 72,79,66,66,89,73,83,84,83,32,68,79,32,73,84,32,66,69,83,84
RUN

HOBBYISTS DO IT BEST

?OD ERROR IN 10

OK

המצא בעצמך את ההודעות המקודדות שלך, או את הדרכים המפוארות לשימוש בפונקציות השונות.

"צילצול מצלצל אחד.."

האם אוזניך מצלצלות? ומה עם המסוף שלך? קיים קליד ושמר BEL, המזוהה עלידי ה-BASIC כמספר ASCII 7. אם אתה משתמש בו בתוך הוראת PRINT, הוא יגרום לצלצול פעמון ברוב המסופים. הקליד BEL חביב ביותר על מתכנני משחקי מחשב, והם משתמשים בו לציון תשובה נכונה או שגיאה, או רק כדי לשמור על העניין. כדי לצלצל בפעמון, נסה את שתי ההוראות הבאות בגישה ישירה.

? CHR\$(7)

אין פלט חזותי, רק צלצול באוזניך.

OK

FOR J=1 TO 8 : ? CHR\$(7) : NEXT J

שמונה צלצולים והכול שפיר. גם שמונה שורות, אחת לכל הוראת PRINT(?).

OK

CHR\$(7)



13 ASCII הוא קליד נוסף העשוי להיות לעתים לתועלת. זהו קליד ההחזרה. בעזרתו תוכל לגרום למחשב להשאיר באותה שורה, ולהדפיס מעל אותה שורה עצמה. פירושו של דבר שהוא מחזיר את המסוף לתחילת השורה, מבלי לעבור לשורה הבאה. הדבר שונה מלחיצה על הקליד RETURN, המעביר לשורה הבאה למטה. האם תוכל לחשוב על שימושים אחרים?

הורה למחשב, (1) להדפיס מחרוזת, (2) להחזיר את המסוף, (3) ולהדפיס מעל אותה שורה מחרוזת אחרת. השתמש בשלוש הוראות בגישה ישירה. (שים לב: הדבר עלול להכשל בסוגי מסופים שונים, אך במקרה זה יש בוודאי דרכים אחרות להזנת ה-מחוג).



```
? "0000000000"; CHR$(13); "/////////"
0000000000
```

OK

לפניכם טבלת קלידי ה-ASCII

מספר	קליד	מספר	קליד	מספר	קליד
000	NUL	043	+	086	v
001	SO11	044	,	087	w
002	STX	045	-	088	x
003	ETX	046	.	089	y
004	EOT	047	/	090	z
005	ENQ	048	0	091	
006	ACK	049	1	092	\
007	BEL	050	2	093	
008	BS	051	3	094	†
009	HT	052	4	095	+
010	LF	053	5	096	~
011	VT	054	6	097	a
012	FF	055	7	098	b
013	CR	056	8	099	c
014	SO	057	9	100	d
015	SI	058	:	101	e
016	DLE	059	;	102	f
017	DC1	060	<	103	g
018	DC2	061	=	104	h
019	DC3	062	>	105	i
020	DC4	063	?	106	j
021	NAK	064	@	107	k
022	SYN	065	A	108	l
023	ETB	066	B	109	m
024	CAN	067	C	110	n
025	EM	068	D	111	o
026	SUB	069	E	112	p
027	ESCAPE	070	F	113	q
028	FS	071	G	114	r
029	GS	072	H	115	s
030	RS	073	I	116	t
031	US	074	J	117	u
032	SPACE	075	K	118	v
033	!	076	L	119	w
034	"	077	M	120	x
035	#	078	N	121	y
036	\$	079	O	122	z
037	%	080	P	123	{
038	&	081	Q	124	
039	'	082	R	125	}
040	(083	S	126	~
041)	084	T	127	DEL
042	.	085	U		

LF=Line Feed

FF=Form Feed

CR=Carriage Return

DEL=Rubout

877

J.R. BROWN	321 MILLBAY ST.	SAN FRANCISCO	CA94123
TIM KELLEY	48 S. AVALON	IOWA CITY	IA52240
M.J. MCPHÉE	1010 DOYLE	MENLO PARK	CA94025
LEROY FINKEL	888 CONSTANCE	MENLO PARK	CA94025
M. SPILANE	98354 SKYLINE	LA HONDA	CA95040

שם כתובת עיר מדינה מיקוד
עד 14 תווים עד 15 תווים עד 16 תווים קיצור בן 5 תווים

DO IT

LIST

5 REM-PRINT ONLY THE MEMBERS NAMES

OK

אויז והחלר



ניתן להפוך מספר למחרוזת או מחרוזת למספר. הפונקציה ההופכת מספר למחרוזת היא STR\$(X). הפיכת מספר למחרוזת פירושה שאין אפשרות להשתמש בצורת המחרוזת לחישובים, אך תוכל להשתמש בכל שאר הפונקציות למחרוזות כדי לטפל במספר שנהפך למחרוזת.

כדי להפוך מחרוזת מספרית למספר, השתמש בפונקציה VAL(X\$).

STR\$(X)

המחרוזת שיש להפוך
לערך מספרי נכתבת כאן.

VAL(X\$)

הערך שיש להפוך
למחרוזת נכתב כאן.

NEW

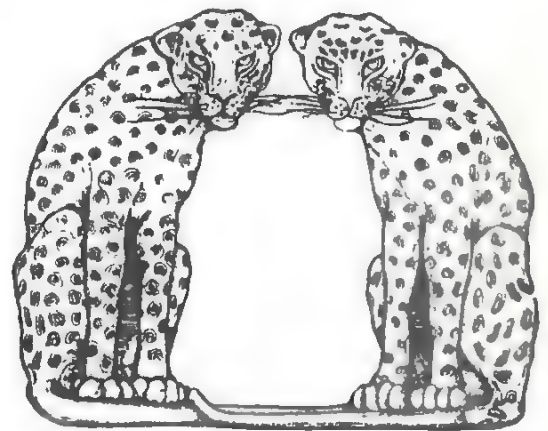
```
OK
10 V=3.14159 : ? V
20 VS=STR$(V) : ? VS
RUN
3.14159
3.14159
```

OK
NEW

```
OK
10 SS="3.14159" : ? SS
20 S=VAL(SS) : ? S
RUN
3.14159
3.14159
```

OK

האם הבחנת במשהו? במחרוזת SS אין רווח לפני המספר עבור הסימן "-" או הסימן "+". כשהמחרוזת SS הופכת לערך, מכניס המחשב רווח זה. (ראה בעמוד הבא.)



בדרך זו תוכל למצוא את אורכו של מספר, כלומר, כמה מקומות דרושים כדי להדפיס את הערך.

NEW

```
OK
10 V=3.14159
20 ? LEN(STR$(V))
RUN
8
OK
```

האם המחשב צודק?
בדוק בעמוד הבא.

ראשית, הפוך את V למחרוזת, ואחר כך מצא את אורכה של המחרוזת.

אי-רווח



זכור ששפת ה-BASIC מותירה רווח בן תו אחד לפני ערך חיובי, עבור הסימן +, למרות שהיא אינה מדפיסה זאת. כשערך חיובי נהפך למחרוזת בעזרת STR\$, הרי שגם המחרוזת כוללת רווח זה. מנה את התווים בפלט שבעמוד הקודם, לצורך בדיקה. זכור שגם הנקודה העשרונית נחשבת לתו. הפיכת ערך למחרוזת מאפשרת לך יתר פיקוח בשאלות כיצד והיכן מודפס מספר זה עלידי המחשב. זאת מפני שאתה יכול ליישם פונקציות מחרוזת לערך שעבר שינוי.

נניח שאנו רוצים להדפיס ערך של משתני FOR-NEXT בזה אחר זה.

NEW

OK

10 FOR K=1 TO 8 : PRINT K; : NEXT K : PRINT

20 FOR K=1 TO 8 : KS=STR\$(K) : PRINT MID\$(KS,2); : NEXT K

RUN

1 2 3 4 5 6 7 8
12345678

OK

איננו רוצים ברווחים אלה כאן. אנו משנים את הערך למחרוזת, ומדפיסים את המחרוזת כשהיא מתחילה בתו השני וזה מבטל את הרווח.

בקרב תתוודע אל יישומים אמיתיים של ערכים שנהפכו למחרוזות. פקח עין כדי שתגלה אותן פונקציות STR\$, המשמשות להפיכת ערכים למחרוזות, כדי לשלוט במיקום הדפסת המספרים.



ועתה, הבה נשתלט על פונקציה מועילה ומעניינת אחרת של BASIC, הנותנת בידך אפשרויות נוספות כדי להשיג את התוצאות הרצויות לך מן המחשב. ככל שתשלוט טוב יותר בחומר, כן ישתכללו תכניות שלך. כל פונקציה חדשה שתלמד, תקרב אותך לניצול המלא ביותר של שפת ה-BASIC ובמחשב שלך.

פונקציה דלת קלוריות, TAB()

השליטה במיקום ההדפסה בעזרת (TAB)



אם תתאמץ, תזכר בוודאי שכבר בתחילתו של ספר זה, סיפרנו לך על קיומם של 72 מקומות בשורה, כמעט בכל המסופים. בוודאי תזכור גם שאמרנו שהמקומות ממוספרים מ-0 ועד 71, כדלהלן: (קרא את המספרים מלמעלה למטה, $\begin{matrix} 1 & 7 \\ 0 & 1 \end{matrix} = 10$ $\begin{matrix} 7 \\ 1 \end{matrix} = 71$):

012345678911111111112222222222333333333344444444445555555555666666666677
01234567890123456789012345678901234567890123456789012345678901

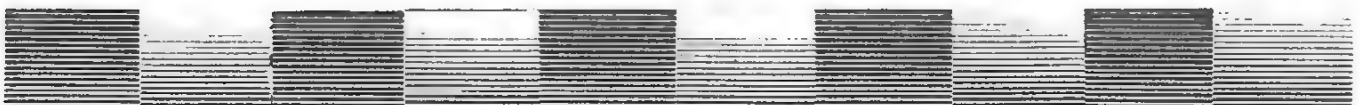
אתה יכול להורות למחשב באיזה מקום תרצה שידפיס משהו, וזאת עלידי השימוש בפונקציה TAB(X). במכונת כתיבה רגילה אתה יכול לקבוע בעזרת הטאבולטור מקומות הדפסה שונים בתוך שורה אחת. הפונקציה TAB ב-BASIC ממלאת תפקיד זהה, ובהוראות PRINT היא משמשת לך כדי לקבוע היכן יודפס כל דבר.

TAB(X)

המקום (0 עד 71) נכתב כאן, ומורה למחשב באיזה מקום עליו להתחיל להדפיס את הפלט הבא. הארגומנט יכול להיות מספר, משתנה או ביטוי שיש לחשב, אך TAB חייב שיהיה לו מספר חיובי, שבין 0 ל-71, כארגומנט שלו. אם הארגומנט הוא מספר בעל שבר, תשתמש הפונקציה TAB רק במספר השלם. לדוגמא, TAB(15.7) מבוצע כ-TAB(15).

בהוראות PRINT (היכן שבדרך כלל משתמשים בה), מופיע אחרי הפונקציה TAB סימן נקודה פסיק, כך שהפלט הכא מתחיל במקום שנקבע עלי ידי ה-TAB.

ואחרון אחרון חביב: הפונקציה TAB אינה יכולה לקבוע את המקומות שמאלה, וכך אם הוא כבר הגיע למקום 45, אינך יכול לחזור למספר 30 או לכל מספר שקודם ל-45.



ה א מ ן () TAB

בתכנית הבאה, אנו משתמשים בהוראת PRINT כדי לצייר ציור או תמונה גראפית של המגדל הנרוטה בפיוזה.

בצע

```
NEW
OK
10 ? "XXX"
20 ? "  XXX"
30 ? "   XXX"
40 ? "    XXX"
50 ? "     XXX"
RUN
XXX
  XXX
    XXX
      XXX
OK
```

ועתה עשה זאת עם TAB.

```
NEW
OK
10 PRINT TAB(0); "XXX"
20 PRINT TAB(1); "XXX"
30 PRINT TAB(2); "XXX"
40 PRINT TAB(3); "XXX"
50 PRINT TAB(4); "XXX"
OK
RUN
XXX
  XXX
    XXX
      XXX
OK
```

ומה דעתך על שרטוט המגדל בדרך אוטומאטית יותר? שים לב לשימוש במשתנה הפיקוח של הלולאה FOR-NEXT לקבלת הנטייה.

```
NEW
OK
10 FOR X=0 TO 4
20 ? TAB(X); "XXX"
30 NEXT X
RUN
XXX
  XXX
    XXX
      XXX
OK
```



היה נכון. היה מקורי. כתוב מספר תכניות שבעזרתן יהפוך מסוף המחשב שלך לאמן גראפי. עם TAB וריבוי הוראות בכל שורה, תוכל להגיע לתכנית די קטנה, שתייצר פלט גראפי גדול. תוכל להשתמש ב-TAB גם כדי לייצר דוחות המורכבים מטורים רבים, המופיעים זה לצד זה.

והרי לפניך שימוש אחר ב-TAB בצירוף מספר פונקציות אחרות, שכבר הספקת ללמוד.
נניח שהינך רוצה לרשום סדרת מספרים בטור, בהתאם למיקום הנקודה העשרונית.

ראשית, המספרים מודפסים כרגיל.

NEW

OK

10 A=1.346 : B=225.1 : C=11.73

20 ? A : ? B : ? C

RUN

1.346

225.1

11.73

OK

אין זה הפתרון המתאים!



זו הדרך לעריכת המספרים בטור, לפי מיקומה של הנקודה העשרונית.

NEW

OK

10 A=1.346 : ? A

20 A1=INT(A) : ? A1 ← טול בעזרת פונקציה INT את חלקי השלם של A (כלומר A עד הנקודה העשרונית בלא השכר).

30 A\$=STR\$(A1) : ? A\$ ← הפוך את הערך למחרוזת, בעזרת הפונקציה STR\$.

40 L=LEN(A\$) : ? L ← מצא את אורכה (LEN) של המחרוזת, דהיינו מספר התווים עד הנקודה העשרונית.

50 ? TAB(6-L) : A ← הפחת את אורכה (LEN) של המחרוזת ממספר כלשהו, נניח 6, כדי לקבל את המקום המתאים בו יש להתחיל בהדפסה, כך שהנקודה העשרונית תגיע תמיד, בסופו של דבר, לאותו מקום. לאחר מכן, יישם את הפונקציה TAB למקום הדפסת התווים, והדפס את ערכו של A. בחרנו באופן שרירותי ב-6 לשימוש בפונקציה TAB. יהיה עליך להמשיך לספור את התווים שלהם תזדקק עבור הטבלה או טור המספרים שלך, או כל דבר שבביצועו תרגלת הסבר זה. ערוך ניסויים!!!

RUN

1.346

1

1

2

1.346

OK

ועתה חבר את כל הפעולות האלו וזכור שכל מה שעליך לעשות, הוא לכתוב סוגריים בזוגות.

NEW

OK

10 A=1.346 : B=225.1 : C=11.73

20 ? TAB(6-LEN(STR\$(INT(A)))) : A

30 ? TAB(6-LEN(STR\$(INT(B)))) : B

40 ? TAB(6-LEN(STR\$(INT(C)))) : C

RUN

1.346

225.1

11.73

חמודי, הכול בסדר, עכשיו הכול כבר בסדר.

OK

109

הקנה דעת לילדיך

האם חבריך, אחיך, אחיותיך וילדיך מקנאים בך על שעומד לרשותך מחשב בו הינך רשאי להשתמש? אנו מביאים כאן תכנית קטנה ונחמדה שתאפשר לילדים לתרגל את שיעורי המתמטיקה שלהם ולשחק עם המחשב, כמו הגדולים.



NEW

OK

```
10 A=INT(RND(1)*10) : B=INT(RND(1)*10)
20 PRINT "A : PRINT "+B : PRINT "---" : INPUT C
30 IF C=A+B THEN PRINT "RIGHT ON!" : PRINT : GOTO 10
40 PRINT "YOU GOOFED, TRY AGAIN." : PRINT : GOTO 20
```

OK

RUN

```
5
+ 5
---
? 10
RIGHT ON!
```

```
9
+ 8
---
? 15
YOU GOOFED, TRY AGAIN
```

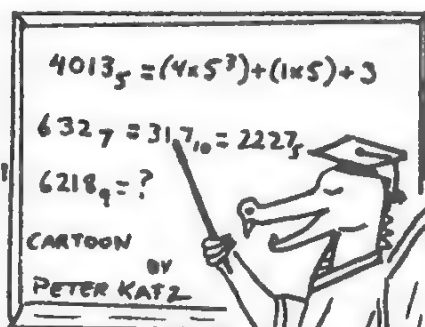
```
9
+ 8
---
? 17
RIGHT ON!
```

```
7
+ 5
---
? 12
RIGHT ON!
```

```
4
+ 4
---
?
```

OK

110



איור זה מופיע במקור בספר
"מה עושים אחרי שלוחצים
על קליד ההחזרה".

ועתה שנה (כתוב והכנס מחדש) שורה 10, כך התכנית תתרגל את חיבור המספרים שבין 0 ל-99.
שנה את שורה 20 כך שתחת המספרים, בבעיית החיבור, יירשמו ארבעה קוקרים או מקפים.



NEW

```
OK
10 A=INT(RND(1)*100) : B=INT(RND(1)*100)
20 PRINT " "; A : PRINT "+"; B : PRINT "--?--" : INPUT C
30 IF C=A+B THEN PRINT "RIGHT ON!" : PRINT : GOTO 10
40 PRINT "YOU GOOFED, TRY AGAIN." : PRINT : GOTO 20
```

OK

RUN

62

+ 66

? 128

RIGHT ON!

13

+ 84

? 97

RIGHT ON!

77

+ 82

? 159

RIGHT ON!

10

+ 4

?

OK



אך ראה, אין זו הדרך בה יופיעו הבעיות בספרי המתמטיקה.

שגוי

10

+ 4

נכון

10

+ 4



יש דרכים רבות לפתרון בעיות תכנות. אפשר להשתמש בפונקציה TAB כדי שתציב את המספרים בבעיות השונות בצורה סטאנדרטית, כשה-1 במקומו וה-10 במקומו, וכך הלאה גם בבעיות של יותר משתי ספרות.

NEW

OK

5 REM-NEW IMPROVED ADDITION PRACTICE PROGRAM

10 A=INT(RND(1)*100) : B=INT(RND(1)*100)

20 AS=STR\$(A) : BS=STR\$(B)

30 ? TAB(5-LEN(AS)); MIDS(AS,2) : ? "+"; TAB(5-LEN(BS)); MIDS(BS,2)

40 ? "----" : INPUT C

50 IF C=A+B THEN ? "RIGHT ON!!!" : ? : GOTO 10

60 ? "YOU GOOFED, TRY AGAIN." : ? : GOTO 30

RUN

78

+ 80

? 158

RIGHT ON!!!

54

+ 18

? 72

RIGHT ON!!!

10

+ 4

? 14

RIGHT ON!!!

5

+ 30

? 35

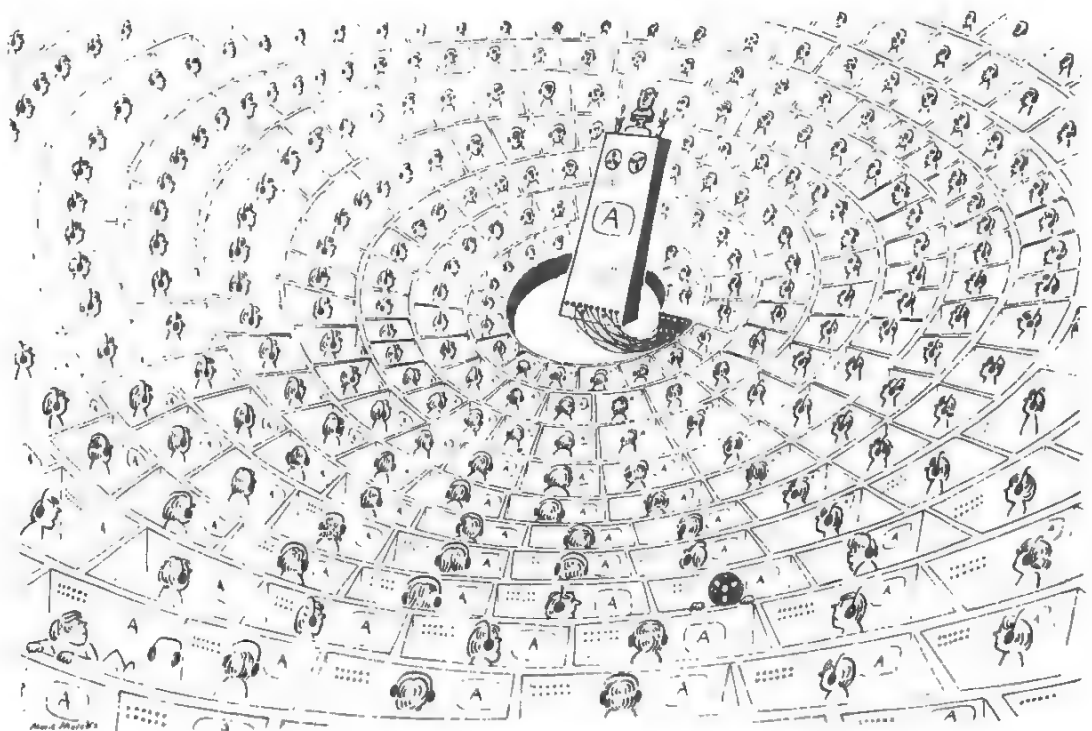
RIGHT ON!!!

47

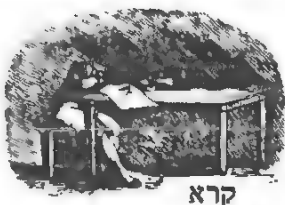
+ 72

? 110

YOU GOOFED, TRY AGAIN.



ציור של מארי מרקס, שהודפס באדיבות חדשות קיזור, 1967



קרא

מיוחד למומחים: (1) אתה יכול לכוון לולאת FSR-NEXT סביב התכנית כדי לספק מספר בעיות במכה אחת. (2) אתה יכול להוסיף הוראת ספירה לשורות 80 ו-90 כדי לעקוב אחר התשובות הנכונות והשגויות, ואף להדפיסן בסוף התירגול. (3) אתה יכול להגביל את מספר התשובות השגויות, לפני שהמחשב מוציא בעיה חדשה, עלידי כריכת לולאת FOR-NEXT אחרת סביב חלקה המרכזי של התכנית. (4) אתה יכול להוסיף הוראות PRINT כדי להסביר כיצד יש להשתמש בתכנית, או אולי לפתור את הבעיות על גבי נייר טיוטא, לפני הכנסת התשובה, במקרה שהן מסובכות. (5) אתה יכול להרשות לילד לבחור מספרים רבים, ככל העולה על דעתו, לכל סוג של בעיה. המשך לחשוב ותכנות נעים!

פונקציות בלתי מוגבלות: הגדר את הפונקציות שלך

אתה יכול ליצור את הפונקציות המיוחדות שלך, באמצעות DEF FN (ראשי התיבות של Define Function - הגדר פונקציה). להלן הדרך בה הינך מגדיר את הפונקציות שלך למחשב.



DEFine FuNction

DEF FNA (X) = "המשתנה המדומה"

ארגומנט משתנה של הפונקציה המוגדרת
משתנה המבדיל פונקציה מוגדרת זו מכל האחרות
המופיעות באותה תכנית.

לאחר שאתה מגדיר פונקציה בהוראה המופיעה בתכנית, אתה רשאי להשתמש בפונקציה בדיוק כמו בכל פונקציה המהווה חלק משפת ה-BASIC. ולמרות זאת, יש כמה תחבולות ומכשלות הקשורות לפונקציות מוגדרות.
(1) עליך להגדיר את הפונקציה שלך בתכנית בטרם תשתמש בה בהוראות אחרות בהמשך התכנית.

(2) משתמשים ב-DEF רק בהוראה שבה הינך מגדיר את הפונקציה.

(3) "המשתנה המדומה" ב-DEF מהווה רק מראה מקום. הוא מראה למחשב היכן בתוך חישוב הפונקציות, יש להחליף את המשתנה המדומה במשתנה בו אתה משתמש הלכה למעשה בארגומנט הפונקציה (בתוך הסוגריים).

(4) אל תתבלבל: המשתנה המזהה את הפונקציה מופיע מיד אחר - FN, היכן שיש לנו חלל ריק. ארגומנט הפונקציה הוא המשתנה המדומה (אנו משתמשים ב-V) בהוראה DEF FN-(V) או המשתנה האמיתי כשהינך משתמש בפונקציה כחלק מהוראה.

(5) אין להשתמש ב-DEF FN עבור מחרוזות או משתני מחרוזות. לצורך המחשה, רבה נחזור לתכנית ששימשה לסידור הנקודות העשרוניות. בטור . אנו מעוניינים בפונקציה מוגדרת שתספק ערך לארגומנט ה-TAB. אנו משתמשים ב-P (POINT) כמשתנה הפונקציה ID.



```
10 A=1.346 : B=225.1 : C= 11.73
20 DEF FNP(V)=6-LEN(STR$(INT(V)))
30 ? TAB(FNP(A)); A
40 ? TAB(FNP(B)); B
50 ? TAB(FNP(C)); C
RUN
```

1.346
225.1
11.73



בוודאי נחסכה כאן
הדפסה רבה!

OK

בכל פעם שהמחשב נתקל ב-FNP, הוא מבצע פעולה זו, כשהוא משתמש בערכו של משתנה הארגומנט במקום במשתנה המדומה. התוצאה היא ערך עבור ארגומנט ה-TAB.

20 DEF FNP(V)=6-LEN(STR\$(INT(V)))

↑ משתנה מדומה
↑ הפונקציה ID (המזהה) משתנה

30 ? TAB(FNP(A)); A

↑ המשתנה האמיתי,
היכן שהיה המשתנה המדומה
↑ פונקציית המשתנה ID

אם הינך מבין את המושג של "הגדרת הפונקציות שלך", כתוב תכנית בה תשתמש ב-DEF FN לצרכיך שלך.

פונקציות טריגונומטריות



האם הטריגונומטריה ושימושיה נכללים בתחום התעניינותך? אם כן, תוכל להסתייע בארבע הפונקציות הבאות:

SIN(X) נותנת את הסינוס של הביטוי X. יחושב ברדיאנים.
הערה: $\text{SIN}(X) = \text{SIN}(X + 3.14159/2)$ ו $\text{COS}(X) = \text{SIN}(X + 180/PL)$ רדיאן, השווה ל-
57.2958 מעלות, כך שהסינוס של X מעלות שווה ל- $\text{SIN}(X/57.2958)$.

COS(X) נותנת את הקוסינוס של הביטוי X. יחושב ברדיאנים.

TAN(X) נותנת את הטנגנס של הביטוי X. יחושב ברדיאנים.

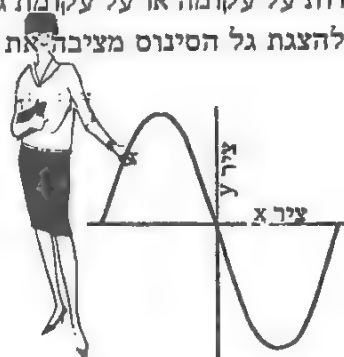
ATN(X) נותנת את הארקטנגנס של הארגומנט X. התוצאה מוחזרת ברדיאנים ונעה בין $(PI/2 - 1.5708) - PI/2 - PI/2$.

במקרה שהטריגונומטריה אינה דווקא השטח שלך, די אם תרשום לפניך את דבר קיומן של פונקציות אלו, אליהן נשוב להתייחס בעתיד. מוטב לך להתרכז בשימוש ב-BASIC להדפסת נקודות המרכיבות עקומה.

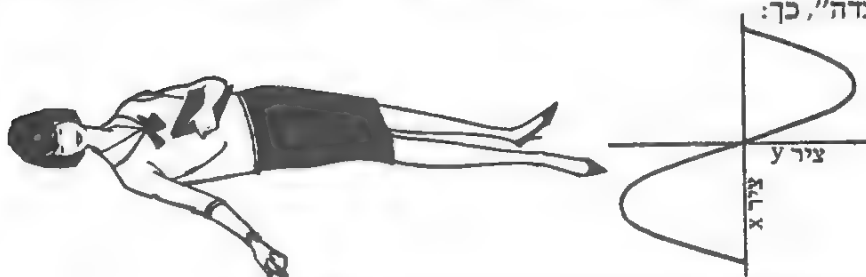


התוכנית המתוחכמת הבאה שלנו ממחישה את השימוש במחשב, לצורך שירטוט עקומה. אנו משתמשים בפונקציה $\text{SIN}(X)$, בביטוי המספק את ערכו של TAB, לצורך הדפסת נקודות על עקומה או על עקומת גל הסינוס.

הדרך המקובלת להצגת גל הסינוס מציבה את הציר X במאונך ואת הציר Y במאונך, כדלקמן:



רוב רובם של מסופי המחשב אינם יכולים לחזור אחורה כדי להדפיס, עובדה ההופכת לבעיה כשהוא מחשב נקודות לאורך עקומה. הפיתרון מתבטא בהדפסת העקומה "הצדה", כך:



בדרך זו יכול המחשב להתחיל "למעלה" ולהדפיס נקודות על העקומה, בזו אחר זו.

לפני יצירת עקומה, עליך להחליט על סולם שיהיה ניתן להדפסה עלידי המחשב וגם יראה את מאפייניה של הפונקציה המתמטית, שתיוצג עלידי העקומה. הואיל ולסינוס של זווית יש ערך הנע בין $+1$ ל- -1 , עלינו להרחיב את הסולם, כדי לשרטט עקומה הנראית כעקומה.



הכפלה ב-10 נותנת לנו סולם של 0 עד 20 על ציר -Y, כלומר 10 רווחים "מעל" ו"מתחת" ציר X האמיתי. פירוש הדבר שיש 20 מקומות הדפסה, בהם ניתן להדפיס נקודות.

$$TAB(10*(1 + SIN(P)))$$

הוספת אחד לערך הסינוס נותנת ערך שבין 0 ו-1, במקום ערך שבין -1 ל-1. הסיבה לכך היא הרצון למנוע ארגומנטים שליליים בפונקציה TAB, הואיל והמחשב אינו יכול להפעיל את TAB לגבי מיקום שלילי. לגבינו, אותם אנשים שאיננו טיפוסים מתימטיים, זה כמו להוסיף +1 כדי לקבל מספר RND שבין 1 ל-10 במקום 0 עד 9 ברוטינת השלם RND, ששימשה אותנו קודם לכן.

FOR P=0 TO 2*3.14159 STEP .3

עלידי החלפת ערך זה, נוכל לפרט כמה מחזורים ישורטטו. $2*PI$ הוא מחזור אחד, כך ש- $4*PI$ הוא שני מחזורים.



בצע

נוכל לציין כמה נקודות נו"ה לרשום לאורך העקומה עלידי שינוי הערך STEP. ככל שערכו של STEP גדול יותר, יתמעט מספר הנקודות שיש לרשום.

```
5 REM-PLOTTING A SINE WAVE
10 FOR P=0 TO 2*3.14159 STEP .3
20 PRINT TAB(20*(1+SIN(P)));"X"
30 NEXT P
OK
RUN
```

```
10 FOR P=0 TO 4*3.14159 STEP .5
RUN
```

מה שינינו כדי לקבל פלט זה?

גירסא ראשונה

גירסא שנייה

שנה את התכנית כדי לראות:

- (1) השפעת שינוי מספר הנקודות שנרשמו.
- (2) השפעת שינוי ה-"גורם הרחבת הסולם"
- (3) השפעת שינוי מספר המחזורים ששורטטו.

OK

OK

הנה מבוא קצר לשתי פונקציות נוספות. $SGN()$ היא קיצור המלה $SIGN$ ונותנת פלוס 1 (+1) או מינוס אחד (-1), תלוי אם הערך שבתוך הסוגריים הוא חיובי או שלילי. $ABS()$ היא קיצור למלה $ABSolute$ (מוחלט) ונותנת לך את הערך שבתוך הסוגריים כמספר חיובי, מבלי להתחשב באם המספר המקורי היה חיובי או שלילי. משתמש כאן בפונקציות INT , כדי שהמחשב יקצץ את השבר העשרוני באותה דרך, מבלי להתחשב באם הערך המקוצץ הוא חיובי או שלילי.

$SGN(B)$ יכול להיות ערך +1 או -1. אם הערך B הוא חיובי אזי $SGN(B) = +1$. אם הערך B שלילי, אזי $SGN(B) = -1$. הפונקציה SGN נותנת למחשב דרך ל"זכירת" סימנו של ערך בעוד הוא מבצע פעולות אחרות עם אותו ערך. היא גם עשויה לסייע בקבלת החלטה בתכנית, המבוססת על העובדה באם B הוא חיובי או שלילי.

$ABS(B)$ נותנת את ערכו המוחלט של מספר $ABS(B) =$ הערך החיובי של B , בין אם בתחילה היה חיובי או שלילי. כדי לקצץ מספר שלילי, אפס, או מספר חיובי, B .

$$A = SGN(B) * INT(ABS(B))$$

$$B = -3.1416 \quad \text{הנח ש-}$$

$$\begin{aligned} A &= SGN(-3.1416) * INT(ABS(-3.1416)) \\ &= -1 * INT(3.1416) \\ &= -1 * 3 \\ &= -3 \end{aligned}$$

פרק 8: בעיות

1. זוכר את שיטת העיגול שפותחה בפרק 6? אפשר וצריך להשתמש בה בפונקציה DEF . כתוב הוראה שתכיל פונקציה כזו ותעשה שימוש בשיטת העיגול.

2. מה תדפיס תכנית זו?

```
10 A$ = "BASIC BEST IS INSTANT DOES"
20 PRINT LEFT(A$,5),
30 PRINT MID(A$,12,2),
40 PRINT MID(A$,7,4)
```

3. כתוב תכנית שתדפיס שורה זו במהופך: $INSTANT BASIC$.

4. הנח שיש לך 20 הוראות $DATA$ המכילות את שמותיהם של 20 מועמדים לעבודה, כששם המשפחה מופיע קודם כשהוא מופרד בפסיק מן השם הפרטי. אתה רוצה להדפיס רשימה של מועמדים כשהשם הפרטי מופיע לפני שם המשפחה. כתוב תכנית שתכין רשימה זו. עליך להשתמש ב- LEN ו- MID יחד עם הפונקציות האחרות.

5. שנה את תכנית תרגול החיבור בפרק זה כך שתאפשר למשתמש בה לחבר, לחסר, להכפיל ולחלק. ולמומחים, שלבו זאת עם הרמזים המיוחדים למומחים המצויים בפרק זה.

6. מהו שם? נולד ילד. המשפחה רוצה בשם בלתי שגרתי, המתחיל באות J , לכבודו של הסבא המנוח, ושלא יכיל יותר מ-6 אותיות. כתוב תכנית שתעלה שמות אקראיים, המתחילים באות J . יהיה עליך להמציא מספר "חוקים" בנוגע למספר התנועות בשם ומיקומן, שאם לא כן תקבל שמות עילגים, אם גם בלתי שגרתיים!

מלכות המשתנים המצוינים



ועתה ברצוננו לחשוף בפניך את סגולתה החשובה האחרונה של שפת ה-BASIC, המקילה על הטיפול המסובך בנתונים ובמידע. זו מלכותם של המשתנים המצוינים, בהם משתמשים כדי לעקוב אחר מידע המופיע ברשימות ובטבלאות. (לא שאתה שולט שליטה מוחלטת בשפת ה-BASIC, אך כל שאר הדברים ששפה זו יכולה לעשות הינם קלים יותר.)



הפרק הדן במשתנים מצוינים יתן בידך כלי חשוב לטיפול במגוון רחב של בעיות תכנות. אם כך, שים לב ואל תחשוש לעקוב אחרי הדוגמאות שלנו ולתרגל אותן, כדי שתרגיש ביטחון בעבודה עם משתנים מצוינים. המתמטיקאים שביניכם עשויים לזכור שמצוינים הם אותם מספרים קטנים (או אותיות קטנות) המופיעים בשיפולי המשתנים בנוסחאות מסובכות, כגון:



החוקים הנוגעים למשתנה המהווה חלק ממשתנה מצויין אינם שונים מאלה הנהוגים ביחס למשתנים רגילים ולמשתני מחרוזת (ראה עמוד.....). שים לב ש- $(3) \times$ הוא משתנה מצויין. יחד עם זאת עליך לשים לב לכך ש- x , $3 \times (3)$, ו- $(3) \times$ הינם משתנים שונים. ניתן להשתמש בכולם במסגרת אותה תכנית. למרות העובדה שהם עלולים לכלבל אותך, הרי שה-BASIC מבדילה היטב ביניהם ויש לה "תאים" נפרדים עבור המשתנים ועבור הערכים או המחרוזות שלהם.

בדומה למשתנה רגיל, מגדיר משתנה מצויין מיקום בתוך המחשב. אנו יכולים לחשוב על מיקומם של תאים קטנים אלה. כך יוכלו התאים להראות, כשיכילו משתנים מצוינים.

A(1)

A(0)	2
A(1)	85
A(2)	3
A(3)	6
A(4)	18
A(5)	-55
A(6)	3.98

קבוצה של משתנים מצוינים, כדוגמת זו, מכונה "טבלה". טבלה "חד מימדית" זו מכונה גם "רשימה" או "קטור". הערך של $A(0)=2$. הערך של $A(1)=85$. הערך של $A(2)=3$ או $A(4)=A(2) \times A(3)$ הסתכל בתאים ומצא את ערכם של $A(2)$ ו- $A(3)$.



ואסור לשכוח גם את הצייני מקום המשתנים

מה שהופך את המשתנים המצויינים האלה לנוחים כל כך, היא יכולתם להקל על הצבת ערכים רבים למשתנים רבים. ומדוע הם מקילים על המלאכה? מפני שהמציין במשתנה מצויין עשוי להיות בעצמו משתנה. (אל בהלה, בדוק בהמשך).

מצייין, שערכו המספרי תלוי בערכו של K . אם $K = 4$, אז התא המתאים $A(K)$ הוא $A(4)$.

K	4
A(4)	18



הבה נוודא כאן אם אנו מבינים שערכו של מצייין אינו הערך המוצב למשתנה מצויין. בהתחלה, עשוי הדבר לכלכל, ולכן נחזור על נקודה זו מספר פעמים. בדוגמא לעיל ערכו של המציין הוא 4 וערכו של המשתנה המצויין $A(4)$ הוא 18. בדוק את התאים. אל תהסס לחזור ולקרוא שנית הקדמה זו. כשתגיע למסקנה שהבנת את התאוריה נסה את תכניות ההמחשה, המשתמשות במשתנים מצויינים.

תכנית זו מציבה ערכים לארבעה ערכים מצויינים מהוראת DATA ולאחר מכן מדפיסה את המשתנים המצויינים ואת הערכים שהוצבו להם.

NEW

OK

5 REM-FIRST SUBSCRIPTED VARIABLE DEMO

10 READ Y(0), Y(1), Y(2), Y(3)

20 ? "Y(0) ="; Y(0)

30 ? "Y(1) ="; Y(1)

40 ? "Y(2) ="; Y(2)

50 ? "Y(3) ="; Y(3)

60 DATA 3, 8, 2, 6.5

RUN

Y(0) = 3

Y(1) = 8

Y(2) = 2

Y(3) = 6.5

OK



ועתה, השתמש בלולאה FOR-NEXT כדי להדפיס את ערכיהם של המשתנים המצוינים. שים לב לשימוש במשתנה הפיקוח A בלולאה FOR-NEXT. בכל מעבר בתוך הלולאה משתמשים בערך של A כדי להדפיס את המציין של המשתנה המצויין. הוא משמש גם כדי להורות למחשב איזה ערך להדפיס עבור המשתנה המצויין. אנו מזהירים אותך שוב ושוב: אל תבלבל בין ערך המציין לבין הערך המוצב למשתנה המצויין.

NEW

```
OK
5 REM-2ND SUBSCRIPTED VARIABLE DEMO
10 READ Y(0), Y(1), Y(2), Y(3)
20 FOR A=0 TO 3
30 ? "Y("; A; ") ="; Y(A)
40 NEXT A
60 DATA 3, 8, 2, 6.5
RUN
Y( 0 ) = 3
Y( 1 ) = 8
Y( 2 ) = 2
Y( 3 ) = 6.5
```

OK

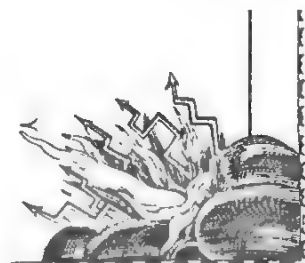


בתכנית ההמחשה הבאה שלנו השתמשנו בלולאה FOR-NEXT כדי להדפיס את המשתנים המצוינים ואת הערכים שהוצבו להם. ועתה, הבה נהפוך גם את הצבת הערכים למשתנים המצוינים לאוטומאטית. תכנית ההמחשה הבאה שלנו משתמשת בלולאה FOR-NEXT כדי שהמחשב יגיע להוראת READ, שתקרא את הערכים מהוראת ה-DATA ותציב אותם, אחד אחד, למשתנה מצויין. ערכו של משתנה הפיקוח של הלולאה FOR-NEXT משמש לקבלת ההחלטה לאיזה משתנה מצויין יש להציב את ערך ה-DATA בשורה 20, ואיזה משתנה מצויין וערכו יש להדפיס בשורה 30.

NEW

```
OK
5 REM-3RD SUBSCRIPTED VARIABLE DEMO
10 FOR A=0 TO 10
20 READ Y(A)
30 ? "Y("; A; ") ="; Y(A)
40 NEXT A
50 DATA 3, 8, 2, 6.5, 211, 81, 1, -32, 7, .3333, 5
RUN
Y( 0 ) = 3
Y( 1 ) = 8
Y( 2 ) = 2
Y( 3 ) = 6.5
Y( 4 ) = 211
Y( 5 ) = 81
Y( 6 ) = 1
Y( 7 ) = -32
Y( 8 ) = 7
Y( 9 ) = .3333
Y( 10 ) = 5
```

OK



הבה נבהיר את העניין עד תומו



שים לב לכך שערכו של A הוא הערך של משתנה הפיקוח של הלולאה FOR-NEXT, הגדל ב-1 (אחד) בכל פעם שהוא עובר בלולאה. על כן, כשהוא עובר בפעם הראשונה בלולאה $Y(A) = Y(0) - 1$ A = 0. הוראת ה-READ מציבה את הערך הראשון, 3, מהוראת ה-DATA לתא המכונה Y(0). אל תבלבל בין ערך משתנה הפיקוח A של הלולאה FOR-NEXT לבין ערכו של המשתנה המצויין Y(A), המוצב מהוראת ה-DATA. אנו פשוט משתמשים בערכו של משתנה הפיקוח A כדי לגרום למחשב להציב ערכים לטבלה או לרשימה של משתנים מצויינים, בזה אחר זה. ערכו של A קובע לאיזה משתנה מצוייין בין Y(0) ל-Y(10) יוצב הערך הבא מתוך הוראת ה-DATA. לולאות ה-FOR-NEXT מקילות בהצבת ערכים רבים למשתנים רבים, ומשתנים מצויינים מקילים על המעקב אחר ערכים רבים.

תכנית בת שורה אחת בלבד

רוצה לתרגל תכנית קטנה אחרונה זו בריבוי הוראות בכל שורה?

עשה זאת כך:

```
10 FOR A=0 TO 10 : READ Y(A) : ? "Y("; A; " ) = "; Y(A) : NEXT A
20   אחרי שהחלפת את שורה 10,
30   הדפס את מספרי השורות שהינך
40   רוצה להוציא ולחץ על RETURN.
LIST
```

```
5 REM-3RD SUBSCRIPTED VARIABLE DEMO
10 FOR A=0 TO 10 : READ Y(A) : PRINT "Y("; A; " ) = "; Y(A) : NEXT A
50 DATA 3, 8, 2, 6.5, 211, 81, 1, -32, 7, .3333, 5
OK
RUN
Y( 0 ) = 3
Y( 1 ) = 8
Y( 2 ) = 2
Y( 3 ) = 6.5
Y( 4 ) = 211
Y( 5 ) = 81
Y( 6 ) = 1
Y( 7 ) = -32
Y( 8 ) = 7
Y( 9 ) = .3333
Y( 10 ) = 5
```

הרץ אותה... בדיק אותן תוצאות

OK (האם קיבלת אותו פלט?)

זו רק בדיקה

העמד את זיכרוןך במבחן. האם אתה זוכר שאחרי הרצת תכנית, כל הערכים מאוחסנים בזיכרון של המחשב? ובכן, השתמש בגישה ישירה כדי לבדוק אם ערכיהם של המשתנים המצויינים זהים לאלה המופיעים בפלט של התכנית האחרונה.



? Y(3), Y(5), Y(10)
6.5 81

5

OK

ובכן, מה אתה אומר? הערכים באמת מאוחסנים בתאים המזוהים על-ידי משתנים מצויינים.

משתני מחרוזת (מצויינים)

גם למשתני מחרוזת יכולים להיות מצויינים. ההבדל היחיד הוא שסימן משתנה המחרוזת (\$) חייב להופיע לפני הסוגריים של המציין, בצורה כזו: Y\$(4). בוא נשתמש בלולאה FOR-NEXT המהימנה שלנו, כדי להציג מחרוזות מהוראת ה-DATA לרשימת משתני מחרוזת מצויינים, Y\$(0) ל-Y\$(6).

NEW



OK

5 REM-SUBSCRIPTED STRING VARIABLE DEMO

10 FOR A=0 TO 6

20 READ Y\$(A)

30 ? "Y\$(" ; A ; ") = " ; Y\$(A)

40 NEXT A

50 DATA APPLE, PEAR, PEACH, CHERRY, PLUM, MELON, GRAPE

RUN

Y\$(0) =APPLE

Y\$(1) =PEAR

Y\$(2) =PEACH

Y\$(3) =CHERRY

Y\$(4) =PLUM

Y\$(5) =MELON

Y\$(6) =GRAPE

OK



לכרמן מירנדה
אין כל קשר
לתכנית זו.

אם אתה זקוק לשיכנוע נוסף, השתמש בגישה ישירה כדי לראות איזו מחרזת מאוחסנת באיזה משתנה מחרזות מצויין.



? Y\$(2), Y\$(4), Y\$(0), Y\$(3)

PEACH

PLUM

APPLE

CHERRY

OK



הבהר זאת היטב לעצמך: אפשר להשתמש במשתנים שונים כדי לקבוע לאיזה משתנים מצויינים מתייחסים. ערכו של המשתנה המשמש לציון המצויין הוא החשוב. בצע הוראה אחרת בגישה ישירה להמחשת נקודה זו. (לא הדפסת NEW, נכון? טוב, אם כן, אל תדפיס זאת!)



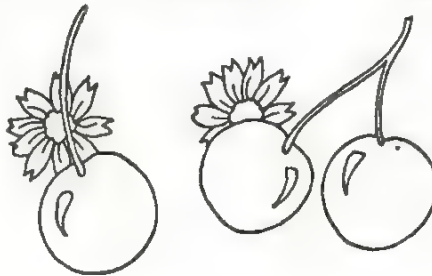
A=3 : B=3 : C=3 : ? Y\$(A), Y\$(B), Y\$(C)

CHERRY

CHERRY

CHERRY

OK



לא משנה איזה משתנה משמש כדי להורות למחשב מהו ערכו של המצויין אם המשתנה המצויין הוא Y\$(3), אזי ידפיס המחשב את המחרזות המוצבת ל-Y\$(3).



הדבר הופך יותר ויותר מעניין ופותח לפנינו עוד ועוד אפשרויות תכנות. כדי לקבוע את ערכו של מצויין יבצע המחשב אפילו חישובים בסוגריים שבתוכם נמצא המצויין. נסה זאת בגישה ישירה, כשרשימת הפירות למשתנים המצויינים Y\$ מאוחסנת עדיין בזיכרוננו של המחשב.

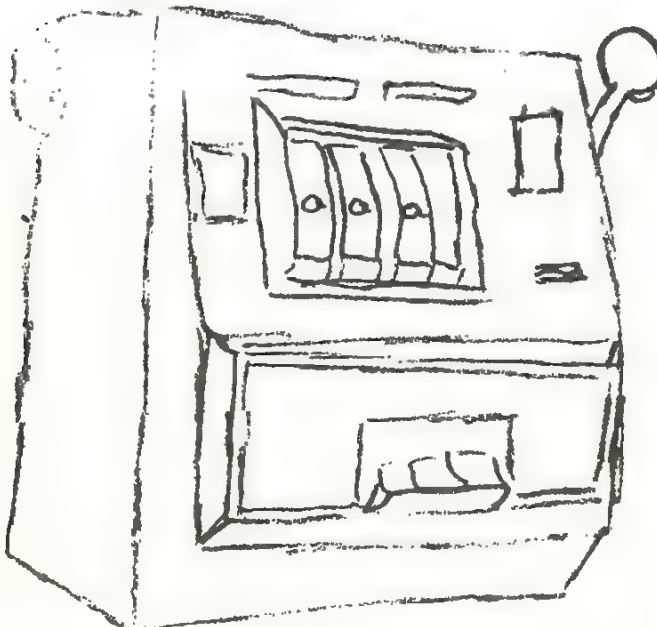
X=2 : Y=8 : Z=6 : ? Y\$(Y/X), Y\$(Z-X), Y\$((Y*Z)/(X*Z))

PLUM

PLUM

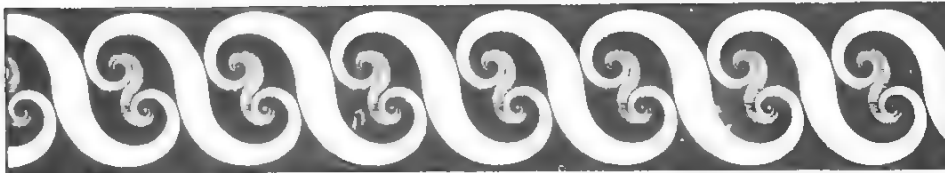
PLUM

OK



נראה שכל אחד מהחישובים בסוגריים של המצויינים נותנים אותו ערך מצויין. האם אתה מסכים? לא היית רוצה שזה יהיה אוטומאט שלתוכו מכניסים מטבעות?

שתי לולאות



בצע

בוא נשתמש עכשיו בשתי לולאות FOR-NEXT שונות; האחת שתקרא את ה-DATA מתוך תאי המשתנים המצויינים והשנייה שתדפיס את תוכנם של תאי המשתנים המצויינים. אם התכנית האחרונה שמורה עדיין בזיכרונו של המחשב (הדפס LIST כדי לוודא זאת, אם רצונך בכך), אתה יכול להמנע מלהדפיס שנית את הוראת ה-DATA באמצעות התחבולה הבאה:

החלף כך את שורה 10.

```
10 FOR A=0 TO 6 : READ Y$(A) : NEXT A
```

ועתה החלף כך את שורה 20:

```
20 FOR B=0 TO 6 : ? Y$(B) : NEXT B
```

השמט שורות 30 ו-40 עלידי הדפסת 30 ואז לחץ על RETURN. הדפס שנית 40 ולחץ שנית על RETURN. זכור, פעולה זו משמיטה שורות אלו מן התכנית, כשהיא מדפיסה במקומן..... לא כלום! ועתה הדפס LIST כדי שתוכל לראות את התכנית המתוקנת, שים לב לשינויים ולאחר מכן הרץ אותה.

```
30
```

```
40
```

```
LIST
```

```
5 REM-SUBSCRIPTED STRING VARIABLE DEMO
```

```
10 FOR A=0 TO 6 : READ Y$(A) : NEXT A
```

```
20 FOR B=0 TO 6 : PRINT Y$(B) : NEXT B
```

```
50 DATA APPLE, PEAR, PEACH, CHERRY, PLUM, MELON, GRAPE
```

```
OK
```

```
RUN
```

```
APPLE
```

```
PEAR
```

```
PEACH
```

```
CHERRY
```

```
PLUM
```

```
MELON
```

```
GRAPE
```

```
OK
```





ועתה רשימת נתונים ארוכה יותר ברשימה או בטבלה \$Y\$. החלף שורות 10 ו-20 והוסף עוד הוראת DATA, בצורה כזו:

```
10 FOR A=0 TO 11 : READ Y$(A) : ? A; Y$(A) : NEXT A
20
60 DATA BANANA, ORANGE, FIG, APRICOT, TURKEY
LIST
```

5 REM-SUBSCRIPTED STRING VARIABLE DEMO

```
10 FOR A=0 TO 11 : READ Y$(A) : PRINT A; Y$(A) : NEXT A
50 DATA APPLE, PEAR, PEACH, CHERRY, PLUM, MELON, GRAPE
60 DATA BANANA, ORANGE, FIG, APRICOT, TURKEY
OK
RUN
```

```
0 APPLE
1 PEAR
2 PEACH
3 CHERRY
4 PLUM
5 MELON
6 GRAPE
7 BANANA
8 ORANGE
9 FIG
10 APRICOT
```



```
?BS ERROR IN 10
OK
```

BS = הם ראשי התיבות עבור מצוין משובש (Bad Subscript). זו הודעת שגיאה אליה טרם התוודענו. בוודאי שאין היא דומה ל-OD ERROR, מפני שיותר ערך נוסף של DATA שעדיין לא נקרא, כש-BS ERROR הופיע בתכנית. זו דרך המקמקה להצגת הנושא הבא.

מימזים חדשים



ובכן מהו מציין משוכש? זהו מציין שהמחשב לא היה מוכן לקבלו. אתה מכין, ה-BASIC מניחה אוטומאטית שהטבלה או רשימת המשתנים המצויינים שלך תכיל לכל היותר מציינים מ-0 עד 10. אך אם ברצונך להשתמש במציינים גדולים יותר, עליך להודיע על כך למחשב. כדי להודיע למחשב את ממדי (DIMensions) הרשימה או הטבלה שלך, השתמש בהוראת ה-DIM. הוראה זו מודיעה למחשב כמה תאים קטנים עליו לשמור עבור המשתנה המצויין, הטבלה או הרשימה.



הערך הגבוה ביותר שאליו עשוי המציין להגיע. ה-BASIC בגירסת Altair מאפשרת שימוש במציינים שבין 0 ל-255. אגא, לא מציינים שליליים, אחרת תקבלו שוב הודעת BS. ואל תנסה להכניס את הוראת ה-DIM יותר מפעם אחת בכל טבלה.



לשם מה כל העניין שאנו עושים מה-DIM הזה? זאת כדי שהמחשב לא ישתמש ברווחים מיותרים עבור משתנים להם אין הוא זקוק במסגרת התכנית שלך. אומרים שטוב להשתמש ב-DIM עבור כל המשתנים המצויינים, גם אם התכנית מכילה פחות מ-11 התאים המותרים ב-BASIC.



ועתה הוסף את הוראת ה-DIM לתכנית הבאה והרץ אותה מחדש.

```
6 DIM Y$(11)
RUN
0 APPLE
1 PEAR
2 PEACH
3 CHERRY
4 PLUM
5 MELON
6 GRAPE
7 BANANA
8 ORANGE
9 FIG
10 APRICOT
11 TURKEY
```

בזכות DIM אנו יכולים להנות מתרגול הדור!



OK

הפעולה היא שווה הן ביחס לערכים והן
ביחס לטבלאות מחרוזות.

ועתה הוסף הוראת DIM ונסה שנית את
התכנית.

NEW

OK

```
10 FOR B=0 TO 15
20 F(B)=B
30 PRINT "F("; B; ") ="; F(B)
40 NEXT B
```

OK

RUN

```
F( 0 ) = 0
F( 1 ) = 1
F( 2 ) = 2
F( 3 ) = 3
F( 4 ) = 4
F( 5 ) = 5
F( 6 ) = 6
F( 7 ) = 7
F( 8 ) = 8
F( 9 ) = 9
F( 10 ) = 10
```

?BS ERROR IN 20

OK

5 DIM F(15)

RUN

```
F( 0 ) = 0
F( 1 ) = 1
F( 2 ) = 2
F( 3 ) = 3
F( 4 ) = 4
F( 5 ) = 5
F( 6 ) = 6
F( 7 ) = 7
F( 8 ) = 8
F( 9 ) = 9
F( 10 ) = 10
F( 11 ) = 11
F( 12 ) = 12
F( 13 ) = 13
F( 14 ) = 14
F( 15 ) = 15
```

OK

תמיד יש מה ללמוד. הוצא מהתכנית האחרונה את שורה 20, כדי לבדוק
מהם הערכים המאוחסנים בטבלה F, במקרה שאינך מציב ערך כלשהו.

20
LIST

```
5 DIM F(15)
10 FOR B=0 TO 15
30 PRINT "F("; B; ") ="; F(B)
40 NEXT B
```

OK

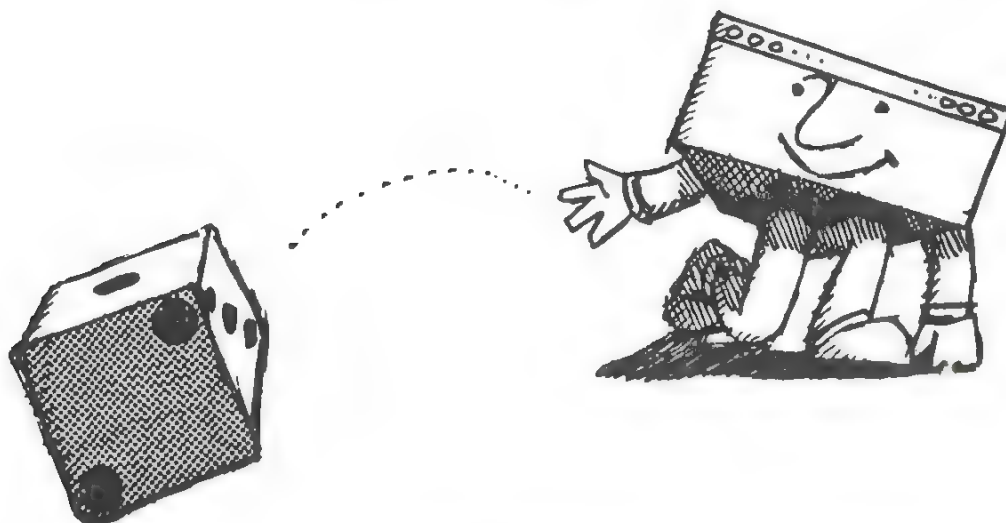
RUN

```
F( 0 ) = 0
F( 1 ) = 0
F( 2 ) = 0
F( 3 ) = 0
F( 4 ) = 0
F( 5 ) = 0
F( 6 ) = 0
F( 7 ) = 0
F( 8 ) = 0
F( 9 ) = 0
F( 10 ) = 0
F( 11 ) = 0
F( 12 ) = 0
F( 13 ) = 0
F( 14 ) = 0
F( 15 ) = 0
```

OK

ה-BASIC מניחה שלכל משתנה,
כולל כל משתנה מצויין, יש ערך 0
עד שמודיעים לה אחרת.

סימולציה של משחקי קוביה



מה שנחמד במשתנים המלווים במציינים הוא, שהמציינים עשויים לשמש לצורך מיון (משתנים או מחרוזות). בו זמנית, עשויה סדרת משתנים מציינים למנות עבורנו ערכים שונים. להלן תרגיל קטן בסימולציה. סימולציה היא חיקוי למצב אמיתי, העשוי להתחרש במציאות. הבה נדמיין זריקת קוביה ונעקוב אחרי המספרים היוצאים בכל זריקה. נעשה זאת עלידי השימוש ב-D כמצוין של משתנה מצוין. ראשית, הכנת הרקע. זוכר את הפונקציה RND? (ראה עמוד 77).

```
30 D=INT(6*RND(1))+1
```

זה יתן לנו זריקת קוביה דמיונית, כלומר, מספר אקראי בין 1 ל-6, המקביל ל-6 פאות של קוביה. ועתה עליך להיזכר כיצד השתמשנו בתכנית זו כדי למנות (ראה עמוד 82).

NEW

OK

10 ? T

20 T=T+1 זוהי הוראת "ספירה".

30 GOTO 10

RUN

0

1

2

3

4

5

BREAK IN 10

OK



ראשית T = 0

T	0
---	---

הערך הראשון של T, מפני שמשנתה שלו לא הוצב כל ערך = 0.

ואז T = 1

T	1
---	---

ערכו של T אחרי שורה 20 מבוצע בפעם הראשונה אחרי הלולאה.

השימוש בטבלה לצורך מעקב

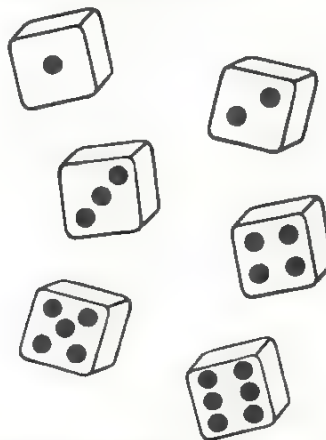
בתכנית הסימולציה נשתמש בטבלה T כדי לעקוב כמה פעמים עולים 1, 2, 3, 4, 5 ו-6 בקוביה הדמיונית שלנו. ראשית, כל הערכים של המשתנה המצויין הם 0.

איננו משתמשים ב-T(0) הואיל ועל הקוביה אין 0.

D	
---	--

ל-D יוצב ערך מתוך הוראת ה-RND בכל פעם שהיא מבוצעת.

T(0)	0
T(1)	0
T(2)	0
T(3)	0
T(4)	0
T(5)	0
T(6)	0



נבנה שההוראה (המופיעה בעמוד הקודם) היוצרת את השלם של RND נותנת לנו 3, כך ש- $D = 3$. בכל פעם ש- $D = 3$ בשורה ה-RND, אנו רוצים שערכו של T(3) יגדל ב-1 כדי להראות את גלגוליו של המספר 3. אנו יכולים להשתמש בהוראת "ספירה" עם המשתנה המצויין של הטבלה T, כדלקמן:

זוהי הוראת "ספירה" $T(D) = T(D) + 1$

If $D = 3$ then $T(3) = T(3) + 1$

ערכו החדש של T(3) ← ערכו הישן של T(3) פלוס 1

D	3
---	---

T(3) = ראשית

T(3)	0
------	---

ערך עכשווי של T(3)

T(3) = 1 ואז

T(3)	1
------	---

אחרי $T(D) = T(D+1)$ מתבצעת $D = 3$

אך אם D יוצא שניים ($D = 2$), אזי $T(2) = T(2) + 1$, כלומר ערכו המאוחסן של T(2) גדל ב-1. הואיל ולקוביה יש 6 פאות עם 1-6 נקודות, נוכל להשתמש בטבלה של 6, כדי לעקוב אחר כל פאה שיוצאת בכל הטלה דמיונית של הקוביה.





ועתה, הבה נאחד את הקטעים. ראשית, עלינו לומר למחשב כמה פעמים עליו "לזרוק את הקוביה".

```
5 REM-SIMULATED ROLLS OF A SIMULATED DIE
10 INPUT "HOW MANY ROLLS"; R
20 FOR K=1 TO R
30 D=INT(6*RND(1))+1
40 T(D)=T(D)+1
50 NEXT K
```

חלק זה מונה את הגלגולים. ←

בסדר. אם אתה מריץ חלק זה של התכנית, יודע המחשב איזה חלק של כל פאה יוצא ב-R זריקות של הקוביה, שכולן מאוחסנות ב"תאים" של הטבלה T. חשוב על דרך בה תביא את המחשב לידי כך שיאמר לך מהם ערכים אלה, או נסה את שיטתנו שבהמשך.

```
60 FOR K=1 TO 6 : PRINT K; "'S:'"; T(K) : NEXT K : PRINT : GOTO 10
```

התינוק זקוק לזוג חדש...

```
OK
5 REM-SIMULATED ROLLS OF A SIMULATED DIE
10 INPUT "HOW MANY ROLLS"; R
20 FOR K=1 TO R
30 D=INT(6*RND(1))+1
40 T(D)=T(D)+1
50 NEXT K
60 FOR K=1 TO 6 : ? K; "'S:'"; T(K) : NEXT K : ? : GOTO 10
RUN
```

HOW MANY ROLLS? 500

```
1 'S: 78
2 'S: 90
3 'S: 99
4 'S: 80
5 'S: 76
6 'S: 77
```



המתנה קצרה בזמן שהמחשב מבצע 500 לולאות ומחשב (מונה) את הפעמים שעולה כל מספר בין 1 ל-6 כשהוא זורק את הקוביה. אפילו מחשב אינו מבצע הוראות רבות כל כך, כהרף עין.

HOW MANY ROLLS? 10000

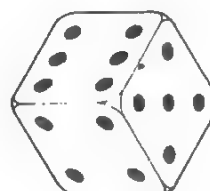
```
1 'S: 1671
2 'S: 1835
3 'S: 1833
4 'S: 1735
5 'S: 1693
6 'S: 1733
```



מרווח זמן מספיק לארוחה קלה או למנוחה קצרה עד שהמחשב מבצע 10,000 לולאות, ואת ההוראות שבתוך הלולאה.

HOW MANY ROLLS?

OK



הקף בעיגול את הבחירה שלך

קרא



דרך זו לשימוש במשתנים המצויינים לצורך ספירה, עשויה להתאים לסוגים שונים של מעקב. שיטת השימוש במצויינים עשויה גם להתאים לסידור תוצאותיו של משאל דעת קהל בתוך טבלה, שאלון, ספירת קולות, תוצאות מבחן או... טוב, חשוב גם אתה על כמה דוגמאות.

אמור שאתה רוצה שהמחשב ימנה את התוצאה לשאלון זה. אפשר היה להטיל עליו גם את ספירת הקולות בבחירות או את התשובות במבחן אמריקאי.



באיזה מועמד תבחר? (הקף בעיגול את המספר המתאים.)

1. זקוק לכמה
2. רוצה יותר

לפניך הקולות בהוראות DATA.

900 DATA 1, 1, 1, 2, 1, 2, 1, 2, 2, 1, 2, 2, 2, 2, 1, 2, 1, 2
910 DATA 1, 2, 2, 2, 2, 1, 2, 1, 1, 1, 2, 2, 1, 1, 2, 1, 2, 2, 2

הבה נבקש מן המחשב שיקראם בזה אחר זה וגם יסדר אותם בטבלה T שלנו. (מין טבלה קטנה שכזו, המכילה רק שני תאים - עבור בחירת 1 ובחירת 2.)

10 READ V

30 T(V)=T(V)+1

T(1)	0
T(2)	0

הערכים הם ראשית כול 0.

V	
---	--

ערכו של V ישתנה בכל קול שנקרא מתוך הוראת ה-DATA.

ערכו החדש של $T(V) = T(V)$ לערכו הישן של $T(V)$ פלוס 1 עבור הקול שנמנה, אם $READ V$ עולה עם $V = 1$ מתוך הוראת ה-DATA, אז

$$T(V) = T(V) + 1$$

או $T(1) = T(1) + 1$

הערך בתא

T(1)	
------	--

או

T(2)	
------	--

גדל ב-1, וכדי לחזק את הנקודה, הדבר תלוי בערכו של מציינ V - בין אם V הוא 1 או שניים במעבר בתוך הלולאה. (עבור לקבלת טיפול הנשמה מפה לפה.)

אחרי שנמנו כל הקולות, נרצה שהמחשב יחזור ויקרא קול נוסף מתוך הוראת ה-DATA.

40 GOTO 10





בוא נאמר שאנו מעוניינים לדעת מה היה מספרם הכולל של הקולות, אך אנו גם מעוניינים שהמחשב יערוך את הספירה למעננו. זכור שה-BASIC אינה יודעת מתי להפסיק כשהיא קוראת קולות (נתונים) מתוך הוראת DATA. אם כך שאנו זקוקים למה שקרוי תמרור שיורה למחשב שכבר הגיע לסופם של הנתונים. זכור איך הוא פועל? כערך האחרון שהוכנס להוראת ה-DATA האחרונה, אתה מכניס ערך השונה לחלוטין מכל האחרים (או מלת קוד עבור נתוני מחרוזת). אז אתה מוסיף הוראת IF-THEN לבדיקת כל ערך לאחר קריאתו, ולקביעה באם הוא ערך תמרור. הבה ניישם את האמור לעיל לתכנית ספירת ומיון הקולות, כשאנו משתמשים ב-9999 בתור תמרור.



```
10 READ V
20 IF V=-9999 THEN 50
30 T(V)=T(V)+1
40 GOTO 10
```



```
900 DATA 1, 1, 1, 2, 1, 2, 1, 2, 2, 1, 2, 2, 2, 1, 2, 1, 2
910 DATA 1, 2, 2, 2, 2, 1, 2, 1, 1, 1, 2, 2, 1, 1, 2, 1, 2, 2, 2
920 DATA -9999
```

ועתה כל מה שאנו צריכים הוא השיטה בה נביא את המחשב לידי כך שיגלה לנו את תוצאות ספירתו.

```
50 ? "TOTAL VOTES:"; T(1)+T(2)
60 ? "CANDIDATE 1:"; T(1) : ? "CANDIDATE 2:"; T(2)
```



בצע

חבר הכול יחד והרץ.



LIST

```
10 READ V
20 IF V=-9999 THEN 50
30 T(V)=T(V)+1
40 GOTO 10
50 PRINT "TOTAL VOTES:"; T(1)+T(2)
60 PRINT "CANDIDATE 1:"; T(1) : PRINT "CANDIDATE 2:"; T(2)
900 DATA 1, 1, 1, 2, 1, 2, 1, 2, 2, 1, 2, 2, 2, 1, 2, 1, 2
910 DATA 1, 2, 2, 2, 2, 1, 2, 1, 1, 1, 2, 2, 1, 1, 2, 1, 2, 2, 2
920 DATA -9999
OK
RUN
TOTAL VOTES: 37
CANDIDATE 1: 16
CANDIDATE 2: 21
```



OK



דו"ח מכירות

הנה דרך שונה במקצת לשימוש בטבלה החד-מימדית. אנו מכנים את התכנית דו"ח מכירות אזורי (Sales Report By Territory). לחברת Peddlers יש שישה אזורים, שבכל אחד מהם פעילים איש מכירות אחד או יותר. כל איש מכירות, מכל אזור, מגיש דו"ח מכירות תלת-חודשי. אנו רוצים לחבר תכנית שתסכם את המכירות לפי אזורים, מבלי להתחשב במספר הדוחות המוגשים עלידי אנשי המכירות, בכל אחד מששת האזורים. כל דו"ח מכירות מוסר: 1) מהו האזור בו פעיל מגיש הדו"ח. (האזורים נושאים מספרים שבין 1 ל-6) 2) מה היה היקף המכירות (בדולרים) של אותו איש מכירות. הוראות ה-DATA שלנו יורכבו מזוגות של נתונים, כשהמספר הראשון מציין את האזור, והמספר השני מציין את היקף המכירות בדולרים.

100 DATA 1,2350, 4,1750, 2,2000, 1,1345, 5,3200, 3,1220, 6,2100
110 DATA 6,1240, 5,2450, 3,4200, 2,1275, 4,1100, 4,1800, 3,900
120 DATA 5,2010, 2,1370, 1,1350, 5,1710, 3,2500, -9999, -9999

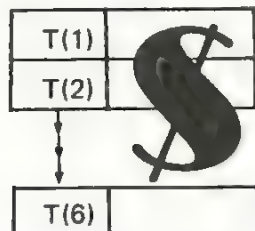


שים לב, שבסיומן הכפול של הוראות ה-DATA מופיעים תמרוורים, מפני שהמחשב יקרא שני ערכים בעת ובעונה אחת, ואנו איננו מעוניינים בהודעת Out of Data, נכון?

ראשית, אנו קובעים את מימדיה של הטבלה המכונה D, ולאחר מכן קוראים את S ו-D, כש-S הוא אזור המכירות ו-D הוא המכירות בדולרים של איש מכירות אחד, הפעיל באותו אזור. אנו משתמשים במציין לצורך מיון המידע אודות המכירות, לפי אזורים וכדי לעקוב אחר הדולרים שנגבו מן המכירות בכל אחד מששת האזורי המכירה (שורה 30).

```
5 REM-SALES REPORT BY TERRITORY
10 DIM T(6)
20 READ S,D : IF D=-9999 THEN 40
30 T(S)=T(S)+D : GOTO 20
```

שים לב ש-T(1) עוקב אחרי המכירות בדולרים באזור 1, T(2) מפקח על המכירות בדולרים באזור 2 וכך הלאה, עד T(6).





ועתה אנו רוצים שהמחשב יסכם עבורנו את כל דוחות המכירות בטבלה שתספק לנו את המידע הרצוי. כשהתכנית מורצת, הרי שהשורות 20 ו-30 מחשבות את המכירות בכל אזור, כשהמציין של T, שהוא מספר בין 1 ל-6, משמש להבחנה בין האזורים. בשלב הבא, אנו זקוקים לכותרת עבור דוח המכירות (שורה 40), למספר האזור ולסך כל המכירות שבוצעו על ידי כל אנשי המכירות הפעילים באזור זה. התשובה מצוייה בלולאה FOR-NEXT. משתנה הפיקוח משמש כדי להדפיס את מספר האזור, וכדי לציין איזה ערך של משתנה מצויין צריך להדפיס (שורה 50).

```
40 PRINT "TERRITORY", "SALES" : PRINT
50 FOR S=1 TO 6 : PRINT TAB(4); S, "$"; T(S) : NEXT S
```

כשאנחנו כבר בשלב זה, בוא נמצא את המספר הכולל של כל המכירות בכל האזורים. הואיל וטרם השתמשנו ב-T(0), נשתמש בו עתה כדי לחבר את כל המכירות, הרשומות ב-T(1) ועד T(6).
שים לב, שהשתמשנו פעמיים ב-TAB כדי למקם טוב יותר את הפלט בתכנית זו.

```
60 PRINT "TOTAL SALES";
70 FOR S=1 TO 6 : T(0)=T(0)+T(S) : NEXT S : PRINT TAB(13); "$"; T(0)
```



בצע

ועתה חבר והרץ הכול, בתנאי שהינך מבין כיצד פועלת התכנית.

NEW

OK



```
5 REM-SALES REPORT BY TERRITORY
10 DIM T(6)
20 READ S,D : IF D=-9999 THEN 40
30 T(S)=T(S)+D : GOTO 20
40 PRINT "TERRITORY", "SALES" : PRINT
50 FOR S=1 TO 6 : PRINT TAB(4); S, "$"; T(S) : NEXT S
60 PRINT "TOTAL SALES";
70 FOR S=1 TO 6 : T(0)=T(0)+T(S) : NEXT S : PRINT TAB(13); "$"; T(0)
100 DATA 1,2350, 4,1750, 2,2000, 1,1345, 5,3200, 3,1220, 6,2100
110 DATA 6,1240, 5,2450, 3,4200, 2,1275, 4,1100, 4,1800, 3,900
120 DATA 5,2010, 2,1370, 1,1350, 5,1710, 3,2500, -9999, -9999
```

OK

RUN

TERRITORY	SALES
1	\$ 5045
2	\$ 4645
3	\$ 8820
4	\$ 4650
5	\$ 9370
6	\$ 3340
TOTAL SALES	\$ 35870

OK



דו"ח משכורות ומכירות

בתכנית הבאה, המכונה דו"ח משכורת ומכירות, אנו מביאים דוגמא, שבצורתה השלמה והמפותחת יותר תמחיש את יישומי המחשב לענייני עסקים, ואתה הרי יודע שבימינו מרבית להשתמש במחשבים בעסקים מכל הגדלים. בתכנית זו נשתמש ב-3 משתנים מצויינים שונים וכן במספר משתנים רגילים, ואפילו במשתנה מחרוזת. התאים עבור המשתנים המצויינים מאחסנים בתוכם מידע הקשור לשמונת הפרטים שחברת Diligent Industries מריצה בעזרת כוח המכירות שלה, המורכב מארבעה אנשים.



כל איש מכירות מוכר אותם 8 פריטים. בכל חודש חייבים האנשים שבכוח המכירות להגיש דו"ח, המראה את הכמות שנמכרה מכל פריט. הדו"ח החודשי יכול להראות כך:

שם								הסניף
1	2	3	4	5	6	7	8	פריט
								מכירות

איש המכירות מקבל משכורת של \$700 בחודש בתוספת 10% עמלה על המכירות שעולות על \$3500.

כמנהל מכירות, הייתי רוצה שהדו"ח הממוחשב שלנו יראה: (1) סך המכירות בדולרים של כל איש מכירות, (2) משכורתו הכוללת של כל איש מכירות, המורכבת מן המשכורת והעמלה. אנו רוצים גם לדעת (3) מהי הכמות שנמכרה מכל פריט, ו- (4) מהי ההכנסה הכוללת, בדולרים, ממכירתו של כל פריט עלידי כל אנשי המכירות, באותו חודש אנו רוצים גם (5) את הסך הכולל של כל המשכורות ששולמו ואת כל ההכנסה מן המכירות של קו היצור של מפעל DI.

כרטיס ספירה

טוב, אינך יכול להבדיל בין המשתנים בלי כרטיס הספירה, המובא בהמשך, כדי לעזור לך להבין כיצד פועלת התכנית ומדוע.

$P()$ מחירים של 8 הפריטים המשווקים על-ידי DI, המוכנסים לתוך תאי המשתנים המצויינים על-ידי שורה 20, הקוראת את הערכים לשורת $P()$ מהוראת ה-DATA הראשונה, במהלך 8 מעברים בלולאה - FOR-NEXT. זכור שהמחירים לכל פריט הם 9 הערכים הראשונים ב-DATA.

$Q()$ ערכים מוכנסים מהוראת ה-DATA, והמספרים המאוחסנים ברשימת $A()$, מבטאים את הכמות של כל אחד מ-8 הפריטים, שנמכרה על-ידי כל איש מכירות. רשימה זו הינה זמנית, מפני שאותו $Q()$ משמש כדי להראות את הכמות שנמכרה משמונת הפריטים על-ידי ארבעת אנשי המכירות (כל אחד בתורו). ראה את הלולאה משורה 90 עד שורה 100 או 110.

$U()$ היחידות שנמכרו משמשות כדי לקיים מעקב אחרי מספר היחידות שנמכרו מכל פריט, בלא להתחשב מי מכר זאת. במלים אחרות, הוא מקיים מעקב אחר המידע המאוחסן באופן זמני ברשימת $Q()$...

X אנו משתמשים בו כמו בלולאות FOR-NEXT (FOR X=1 TO 8...NEXT) כדי לקרוא את הערכים מהוראות ה-IATA, ולתת את מקום המצוין עבור ערכים אלה, כלומר $P(X)$, $Q(X)$ ו- $U(X)$.

S מחשב את $S4$ - המכירה הכוללת, בדולרים, של כל אנשי המכירות.

$S1$ מחשב את סך המשכורות ששולמו לאנשי המכירות.

$S2$ הוא הסך הכולל של המכירות בדולרים.

$S3$ הוא משכורתו של איש המכירות שמכר פריטים בסכום של 3500 דולרים, כפי שמשקף מדו"ח המכירות החדשי שלו.

$S5$ מחשב את סך המכירות לאיש, כל אחד בתורו, כך שצריך להחזירו לאפס, לאחר שהוכנסו מכירותיו של כל אחד מאנשי המכירות (ראה שורה 40), אך הוא גם מחשב את הערך הדולרי של הפריטים שנמכרו (שורה 80) ולפני שהוא מוחזר לאפס, מוסיפים את הערך שחושב ל-S, כדי לקבל את סך המכירות, בדולרים, לחודש.

NS משמש לשמירת שמותיהם של אנשי המכירות, כדי להדפיסם בשורה המסכמת את מכירותיהם ומשכורתם, ולקרוא מהוראת ה-DATA, את מספר הפריטים שנמכרו על-ידי כל איש מכירות.

המשתנים



NEW

OK

```

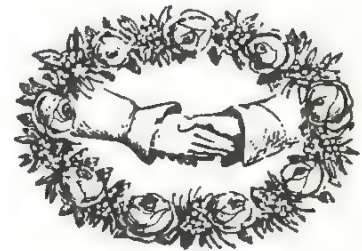
10 DIM P(8), V(8), T(8), Q(8), NS(12)
20 FOR X=1 TO 8 : READ P(X) : NEXT X
30 PRINT "SALESPERSON", "TOTAL SALES", "SALARY"
40 S4=0
50 READ NS : IF NS="END" THEN 120
60 FOR X=1 TO 8 : READ Q(X)
70 U(X)=U(X)+Q(X)
80 S4=S4 + Q(X)*P(X) : NEXT X
90 PRINT NS, S4,
100 S=S+S4 : IF S4<=700 THEN PRINT 700 : GOTO 40
110 S3=700+((S4-3500)*.1) : PRINT S3 : S1=S1+S3 : GOTO 40
120 PRINT : PRINT "TOTALS", S, S1 : PRINT
130 PRINT "ITEM", "PRICE/ITEM", "UNITS SOLD", "TOTAL SALES"
140 FOR X=1 TO 8 : PRINT X, P(X), U(X), U(X)*P(X)
150 S2=S2+U(X)*P(X) : NEXT X
160 PRINT : PRINT "GRAND TOTAL OF SALES",, S2
200 DATA 2.05, 18.45, 6.75, 9.95, 25.00, 16.50, 5.50, 12.60
210 DATA D.MILLER, 120, 15, 75, 0, 20, 100, 80, 144
220 DATA B.MIDLER, 160, 1, 90, 55, 16, 120, 96, 132
230 DATA P.PADRE, 80, 10, 60, 40, 5, 75, 10, 55
240 DATA A.XAVIER, 144, 60, 96, 96, 36, 144, 106, 90
250 DATA END

```

OK

RUN

SALESPERSON	TOTAL SALES	SALARY
D.MILLER	5433.4	893.34
B.MIDLER	6072.4	957.24
P.PADRE	3262	676.2
A.XAVIER	7998.4	1149.84



TOTALS	22766.2	3676.62
--------	---------	---------

ITEM	PRICE/ITEM	UNITS SOLD	TOTAL SALES
1	2.05	504	1033.2
2	18.45	86	1586.7
3	6.75	321	2166.75
4	9.95	191	1900.45
5	25	77	1925
6	16.5	439	7243.5
7	5.5	292	1606
8	12.6	421	5304.6

GRAND TOTAL OF SALES

22766.2

OK

פרק 9 - בעיות

1. ענה על השאלות הבאות, הקשורות לתכנית דו"ח המכירות והמשכורת שבפרק זה.

(א) באיזו הוראה מחברים את כל היחידות שנמכרו? _____

(ב) מה קורה בשורה 90? _____

(ג) שורה 100 כוללת $S = S + S4$. מה פירושו של דבר? _____

(ד) באיזו הוראה ממוחשבת המשכורת? _____

2. 5 FOR X = 1 TO 5

10 READ A\$(X), S(X)

20 NEXT X

30 DATA D. MILLER, 500, B. MIDLER, 950, P. PADRE, 1400

40 DATA J. BLACK, 1500, L. FRENCH, 2000

מה יימצא במשתנים אלה, אחרי שתכנית זו תורץ?

A\$(5) _____

A\$(2) _____

S(3) _____

S(1) _____

3. 10 FOR N = 1 TO 8; READ A(N); NEXT N

20 DATA 3, 5, 2, 16, 4, 5, 9, 7

מה יהיה הערך במשתנים אלה אחרי הרצת תכנית זו?

A(4) _____ A(7) _____ A(A(1)+A(5)) _____ (הזהר)

A(3*A(3)) _____

4. מתי צריך להשתמש בהוראת ה-DIM?

5. מועדון המחשבים המקומי מבקש ממך להכין תכנית, שתמנה את הקולות בבחירות הבאות לבעלי תפקידי המפתח במועדון. הנה הקלפי. כתוב את התכנית שתצבור קולות ותדפיס דו"ח נאה (עשה עבודה טובה שתרשים את כל יתר החברים!)

נשיא	סגן נשיא	גזבר
1. A. ABLE _____	1. M. MAC _____	1. S. SOBER _____
2. B. BAKER _____	2. J. ROVER _____	2. T. MILSTON _____
3. C JONES _____	3. M MANGI _____	3. G GONIF _____

עליך לסדר את התוצאות של כל הצבעה בהוראת DATA, כדלקמן:

DATA 2, 1, 3, 3, 2, 3

6. לפניך מונח גיליון ציונים. במכללה בה בחרת ללמוד, מקבלים כל תלמיד בעל 75 ציוני A ו-B. כתוב תכנית שתמנה את ציוני ה-A וה-B שלך. הכנס את כל ציוניך בהוראת DATA, כשאתה משתמש בסולם זה:

$A = 4 \quad B = 3 \quad C = 2 \quad D = 1 \quad F(n/c) = 5$

DATA 4, 3, 4, 2, 2, 1, 4, 3

האוטומאט להכנסת מטבעות מפורשט. תרגיל טבעת המחירות שבפרק זה, עם דובדבנים ושזיפים, יוצר מצב טבעי לעריכת סימולציה של אוטומאט להכנסת מטבעות (בנוסף נבאדה). נסה לכתוב תכנית העורכת סימולציה של מכונה פשוטה, אך שכלל אותה בהגדלת סיכויי הרווח באמצעות הוספת סכומי כסף גדולים יותר.

צורה פשוטה. קרא מתוך הוראת DATA לטבלה: BAR, CHERRY, APPLE, PLUM, PEACH. אחר כך בחר שלוש אפשרויות באופן שרירותי והדפס התוצאות.

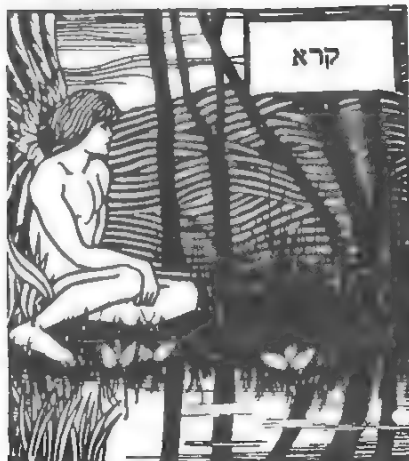
אם טור 1 הוא cherry, אתה מרוויח 0.05 דולר.

אם טור 1 ו-2 זהים, אתה מרוויח 0.10 דולר.

אם טור 1, 2, ו-3 זהים, אתה מרוויח 0.25 דולר.

אם טור 1, 2, 3 הם BAR אתה מרוויח כל מה שנמצא בבטן המכונה.

2- מנה אותם -2



אם דהרת בין פרקיו של ספר זה, ייתכן שזהו הזמן המתאים להפוגה. חשוב שתרגיש שהשתלטת בקלות על נושא המשתנים המצויינים. האם ניסית לתרגל את התכניות לדוגמא? האם הינך מבין כיצד מאחסנים המשתנים המצויינים את הערכים? האם ניסית ליישם רעיונות משל עצמך, המשתמשים במשתנים מצויינים? אם אתה זקוק לחזרה, זהו הזמן, מפני שאנו עומדים להנחית עליך צורה מורחבת של משתנים מצויינים.



עד כה השתמשנו במשתנים מצויינים כדוגמת $P(8)$ ו- $T(D)$ כבעלי מצויינים בודדים, כלומר לכל משתנה יש רק משתנה בודד אחד.

$P(8)$ $T(D)$ $Y\$(X)$

עתה יש לנו משהו חדש בשבילך, משתנים מצויינים כפולים, כלומר משתנים בעלי מצויינים כפולים.

$P(2,3)$ $Y\$(2,5)$ $T(A,B)$

המצויינים מופרדים עלידי פסיק. דרך פשוטה לתיאור משתנים בעלי מצויינים כפולים היא בעזרת תאים קטנים, המסודרים בטבלה של שורות וטורים, כך:

אנו מציגים את המשתנה שלנו בעל המציין הכפול, $A(4,3)$.

	טור 1		טור 2		טור 3	
שורה 1	$A(1,1)$	0	$A(1,2)$	0	$A(1,3)$	0
שורה 2	$A(2,1)$	0	$A(2,2)$	0	$A(2,3)$	0
שורה 3	$A(3,1)$	0	$A(3,2)$	0	$A(3,3)$	0
שורה 4	$A(4,1)$	0	$A(4,2)$	0	$A(4,3)$	0

התאים מראים שכל הערכים המוצבים למשתנים בטבלה דרמימדיית זו, הם אפס.

$A(4,3)$

סידור מוסכם זה מאפשר לנו לייחס את המציינים למקומות או ל"תאים" מסוימים, המציינים ערכים בשורות ובטורים. מעוניין בכמה מונחים? הסידור המלבני של משתנים בעלי מציין כפול בשורות ובטורים מכונה טבלה, או מטריצה, או טבלה דרמימדית. זכור שטבלה של משתנים בעלי מציין בודד מכונה רשימה או טבלה חד-מימדית.

זוהי רשימה או טבלה חד-מימדית.

B(0)	
B(1)	
B(2)	
B(3)	

מציין אחד, מימד אחד



זוהי טבלה או טבלה דרמימדית:

A(0,0)		A(0,1)		A(0,2)		A(0,3)		A(0,4)		A(0,5)	
A(1,0)		A(1,1)		A(1,2)		A(1,3)		A(1,4)		A(1,5)	
A(2,0)		A(2,1)		A(2,2)		A(2,3)		A(2,4)		A(2,5)	
A(3,0)		A(3,1)		A(3,2)		A(3,3)		A(3,4)		A(3,5)	
A(4,0)		A(4,1)		A(4,2)		A(4,3)		A(4,4)		A(4,5)	

שני מציינים
שני מימדים

בדיוק כמו למשתנים המצויינים הבודדים, גם אצל בעלי המציינים הכפולים עשויים המציינים להתחיל מאפס. לטבלה זו ניתן להציב עד 30 ערכים או מחרוזות, כשכל אחד מזוהה על-ידי המשתנה המצויין שלו. בודאי שאם הייתה זו טבלת מחרוזות, היה מופיע סימן \$ אחרי ה-A, כך: A\$(3,4). טבלאות שנוצרו על-ידי השימוש במציינים כפולים הן נוחות לאיחסון נתונים בתוך המחשב. אתה חייב לומר למחשב שלך מהם המימדים - (DIMensions) המירכיים של המשתנים שלך, בעלי המציין הכפול. אתה רשאי להשתמש באותה הוראת DIM, כדי לכלול את כל המשתנים בהוראה אחת, בין אם הם משתני מחרוזות, או משתנים בעלי מציין בודד או כפול.

20 DIM C(3,4), X\$(15), Y(12), M\$(F,S)

מספר שורה ↑

טבלה חד-מימדית טבלת מחרוזות חד-מימדית טבלה חד-מימדית טבלת מחרוזות חד-מימדית

תופס את הרעיון? משתנה בעל מציין כפול הוא פשוט שמו של תא או מיקום בתוך המחשב, בהם אתה יכול לאחסן ערך או מחרוזת, בדיוק כמו המשתנים האחרים בהם השתמשת. בדיוק כמו שלמשתנה בעל מציין בודד, יכול להיות מציין שלמעשה הינו משתנה, גם למשתנה בעל מציין כפול יכולים להיות שני מציינים שהינם משתנים, או אפילו ביטויים שיש לחשב. וודאי שלמחשב חייבת להיות דרך כלשהי לידיעת ערכם של המשתנים המשמשים כמציינים. לצורך תרגול, מלא את הרווחים בתאים עבור המשתנים המצויינים, עם ערכי המציינים.

$$B(K,2) = 365$$

$$K = 8$$

$$C(R,C) = 4$$

$$R = 0$$

$$C = 3$$

$$F\$ (N-1, K*2) = \text{"HOT"}$$

$$N = 3$$

$$K = 2$$

K	8
B(____,2)	365

K

R	0
C	3
C(____,____)	4

R

C

שים לב, שערכו
המספרי הבודד של
C שונה מהמשתנה
המצויין C(R,C).

N	3
K	2
F\$(____,____)	HOT

N-1

K*2

המחשת המציין הכפול

בצע

NEW

OK

5 REM-DOUBLE SUBSCRIPT DEMO

10 DIM C(2,3)

20 READ C(0,0), C(0,1), C(0,2), C(0,3)

30 READ C(1,0), C(1,1), C(1,2), C(1,3)

40 READ C(2,0), C(2,1), C(2,2), C(2,3)

50 ? C(0,0), C(0,1), C(0,2), C(0,3)

60 ? C(1,0), C(1,1), C(1,2), C(1,3)

70 ? C(2,0), C(2,1), C(2,2), C(2,3)

100 DATA 5, 10, 88, -19, 100, 8.25, 91, 22, -1.5, 15, 9, 2

RUN

5	10	88	-19
100	8.25	91	22
-1.5	15	9	2

OK

ועתה מלא את הערכים שהוצבו למשתנים מצויינים אלה בתכנית האחרונה. אנחנו הסתפקנו במילוי שני הראשונים.

C(0,0)	5	C(0,1)	10	C(0,2)		C(0,3)	
C(1,0)		C(1,1)		C(1,2)		C(1,3)	
C(2,0)		C(2,1)		C(2,2)		C(2,3)	

ועתה, בוא נתחיל באוטומאציה. כפי שעשינו קודם לכן עם טבלאות חד מימדיות, אנו משתמשים בלולאות FOR-NEXT כדי לתת ערכים למצויינים, ובדרך זו מורים למחשב באיזה משתנה מצויין עליו לטפל. אך עתה יש לנו שני מצויינים, כך שאנו משתמשים בלולאות מוכנסות FOR-NEXT כדי להעביר בהן את הערכים האפשריים.

בתכנית האחרונה החלף את השורות 50, 60 ו-70 והוסף שורה 80.

בצע

```
50 FOR R=0 TO 2
60 FOR C=0 TO 3
70 ? "C(" ; R ; ", " ; C ; ") = " ; C(R,C) ; " " ;
80 NEXT C : ? : NEXT R
LIST
```

```
5 REM-DOUBLE SUBSCRIPT DEMO
10 DIM C(2,3)
20 READ C(0,0), C(0,1), C(0,2), C(0,3)
30 READ C(1,0), C(1,1), C(1,2), C(1,3)
40 READ C(2,0), C(2,1), C(2,2), C(2,3)
50 FOR R=0 TO 2
60 FOR C=0 TO 3
70 PRINT "C(" ; R ; ", " ; C ; ") = " ; C(R,C) ; " " ;
80 NEXT C : PRINT : NEXT R
100 DATA 5, 10, 88, -19, 100, 8.25, 91, 22, -1.5, 15, 9, 2
OK
```

```
80 NEXT C : ? : NEXT R
```

משמש כדי "לבטל"
את סימן ; בסופה של
שורה 70 ולהתחיל
שורת פלט חדשה.

האם התקשית בהכנסת שורה 70? הבה נתבונן בה ביתר תשומת לב.

70 ? "C(" ; R ; ", " ; C ; ") = " ; C(R,C) ; " " ;

השאר רווחים
בין המשתנה
המצויין לערכים.
הישאר
באותה
שורה

הסוגריים של
המצויין נסגרו.
הדפס את הפסיק
המבדיל בין
המצויינים.

החלקו הראשון של רישום
המשתנה המצויין

הדפס את
הערך המוצב
למשתנה
זה.

ערכו של
המצויין השני.

ערכו של
המצויין הראשון.



ועתה הרץ את התכנית.

RUN

C(0 , 0) = 5 C(0 , 1) = 10 C(0 , 2) = 88 C(0 , 3) = -19
C(1 , 0) = 100 C(1 , 1) = 8.25 C(1 , 2) = 91 C(1 , 3) = 22
C(2 , 0) = -1.5 C(2 , 1) = 15 C(2 , 2) = 9 C(2 , 3) = 2

OK



טוב, אני חושב שכבר ניחשת זאת. השלב הבא הוא להפוך את הצבת הערכים למשתנים בעלי מציין כפול לאוטומאטית. אם כך, בוא נשנה שוב את התכנית, שתעבור אוטומאטית בטבלה C ותציב את הערכים לאותם משתנים מצויינים.

בצע

החלף שורות 20, 30 ו-40 בלולאות FOR-NEXT מוכנסות אלו.

```
20 FOR R=0 TO 2 : FOR C=0 TO 3
30 READ C(R,C)
40 NEXT C,R
LIST
```

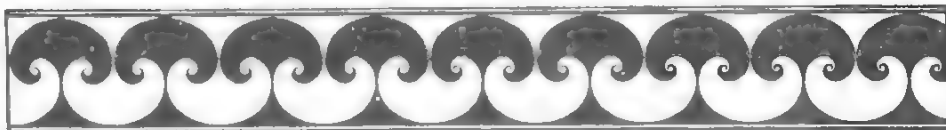
```
5 REM-DOUBLE SUBSCRIPT DEMO
10 DIM C(2,3)
20 FOR R=0 TO 2 : FOR C=0 TO 3
30 READ C(R,C)
40 NEXT C,R
50 FOR R=0 TO 2
60 FOR C=0 TO 3
70 PRINT "C("; R; ", "; C; ")="; C(R,C); " "
80 NEXT C : PRINT : NEXT R
100 DATA 5, 10, 88, -19, 100, 8.25, 91, 22, -1.5, 15, 9, 2
```

OK

RUN

C(0 , 0) = 5 C(0 , 1) = 10 C(0 , 2) = 88 C(0 , 3) = -19
C(1 , 0) = 100 C(1 , 1) = 8.25 C(1 , 2) = 91 C(1 , 3) = 22
C(2 , 0) = -1.5 C(2 , 1) = 15 C(2 , 2) = 9 C(2 , 3) = 2

OK



השבוע עכ



בוא נחזור לתכנית ספירת הקולות שלנו. השימוש במשתנים בעלי מציין כפול מאפשר לנו להוסיף עוד מימד לעריכת החשבון או הספירה. השאלון החדש שלנו מקשה: "למען מי הצבעת בבחירות הקודמות?"



שאלה 1: למען מי הצבעת בבחירות הקודמות? הקף את מספר תשובתך בעיגול	שאלה 2: הקף בעיגול את הגיל שלך.
1. איש הכוח	1. 18 - 29
2. איש הממון	2. 30 - 39
3. אחר	3. 40 - 49
	4. 50 או יותר



ברצוננו לכתוב תכנית שתסכם את הנתונים שנאספו בבחירות אלו. שים לב לכך שיש שתי שאלות, כשלאשונה יש 3 תשובות אפשריות ולשנייה 4 תשובות אפשריות כל שאלון נותן שתי תשובות. תשובה אחת לשאלה 1 (העשוייה להיות 1, 2 או 3) והתשובה לשאלה 2 (העשוייה להיות 1, 2, 3 או 4). ברור שאנו רוצים לדעת את התשובות ביחס לקבוצות הגיל, בדיוק כמו מכוני המשאלים הגדולים.

זוכר כיצד השתמשנו במציין של משתנה כדי לקבוע באיזו קבוצה לשבץ את הקול?

10 READ V
30 $T(V) = T(V) + 1$

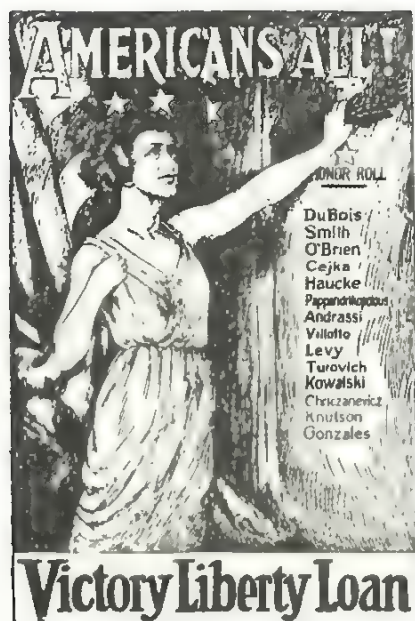


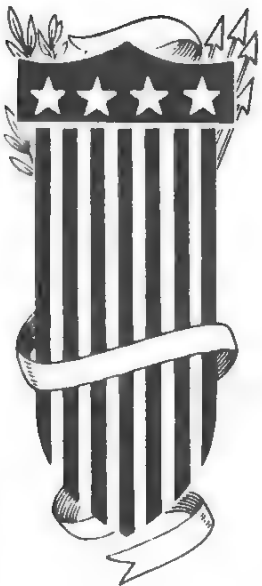
T(1)	
T(2)	



עתה עלינו לחשב שתי תשובות, ולכן אנו נעזרים בטבלה דרמימדית.

	18 - 29	30 - 39	40 - 49	50 או יותר
איש הכוח	C(1,1)	C(1,2)	C(1,3)	C(1,4)
איש הממון	C(2,1)	C(2,2)	C(2,3)	C(2,4)
אחר	C(3,1)	C(3,2)	C(3,3)	C(3,4)





הואיל ויש שתי שאלות, מורכב ה-DATA שלנו לכל תשובה משני מספרים, שהם התשובה לשאלה 1 (אנו משתמשים ב-V לציון ההצבעה) והתשובה לשאלה 2 (אנו משתמשים ב-A לציון קבוצת גיל).

יש 12 צירופי תשובות שונים. הצבעה עבור 1 (עבור איש הכוח) על-ידי מישו מקבוצת הגיל 1 (18 - 29) נותנת לנו את צמד התשובות 1, 1. הצבעה עבור 3 (אחר) על-ידי מישו מקבוצה 2 (30 - 39) נותנת לנו את צמד התשובות 2, 3. האם אתה מתחיל להבין כיצד משתנה בעל מציין כפול בטבלה C (V,A) עשוי לסייע בספירת 12 הצירופים האפשריים לשאלון?

הואיל ויש 3 תשובות אפשריות לשאלה 1, ו-4 קבוצות גיל אפשריות לכל משיב, אנו זקוקים לטבלה של 3 על 4, ואנו נתחיל את התכנית בהוראת DIM.

10 DIM C(3,4)

נקרא A, V מהוראות ה-DATA. רק למען הבהרת העניין, נקדד את התשובות לשאלונים בזוגות מספרים בהוראות ה-DATA. להלן התשובות לשאלון שלנו. זכור שכל תשובה מורכבת מזוג מספרים.

900 DATA 1,2, 1,3, 2,2, 2,3, 2,1, 3,2, 3,4, 2,3, 2,3, 3,2, 1,4
910 DATA 3,1, 3,2, 1,4, 2,4, 2,3, 1,3, 2,4, 1,4, 2,2
920 DATA 2,1, 2,2, 3,2, 2,4, 1,2, 1,3, 2,4, 1,4, 2,3, 3,1, 3,3

אנו זקוקים לתמרוך לסימן סוף ה-DATA. יחד עם זאת, אנו קוראים שני ערכים ברזמנית (READ V, A) מן ה-DATA, אנו זקוקים לשני תמרוכים, כך שהמחשב לא ישדר לנו הודעת OD ERROR. כמו כן, אנו נזקקים גם להוראה לחפש את התמרוך ולהורות למחשב היכן להסתעף אחרי שסיים את קריאת כל ה-DATA.

20 READ V,A : IF A=-9999 THEN 40
930 DATA -9999, -9999

טבעי שאנו רוצים שהמחשב יערוך את הספירה. אם כך, אנו זקוקים להוראה לחשב את התשובה לכל שאלון שבתא, המייצג כל משתנה מצויין. האם אתה זוכר שכל תשובה הינה צמד תשובות - ההצבעה וקבוצת הגיל. ואז אנו רוצים שהמחשב יסתעף חזרה ויקרא את צמד הערכים הבא.

30 C(V,A)=C(V,A)+1 : GOTO 20





עד כה, חלקה זה של התכנית (בתוספת הוראות ה-DATA) ימנה את כל התשובות לשאלונים.

LIST

5 REM-VOTE COUNTING WITH TWO-DIMENSIONAL ARRAY

10 DIM C(3,4)

20 READ V,A : IF A=-9999 THEN 40

30 C(V,A)=C(V,A)+1 : GOTO 20

900 DATA 1,2, 1,3, 2,2, 2,3, 2,1, 3,2, 3,4, 2,3, 2,3, 3,2, 1,4

910 DATA 3,1, 3,2, 1,4, 2,4, 2,3, 1,3, 2,4, 1,4, 2,2

920 DATA 2,1, 2,2, 3,2, 2,4, 1,2, 1,3, 2,4, 1,4, 2,3, 3,1, 3,3

930 DATA -9999, -9999

OK



החלק הבא של התכנית חייב להורות למחשב לומר לנו מהן תוצאות הספירה שערך. אנו רוצים במידע הבא בעת הרצת התכנית.

RUN CANDIDATE	18-29	30-39	40-49	50 +
POWERMAN	0	2	3	4
MONEYMAN	2	3	5	4
OTHER	2	4	1	1

OK

ראשית, הוראה שתדפיס את הכותרות ושורה ריקה.

40 ? "CANDIDATE", "18-29", "30-39", "40-49", "50 +" : ?

ועתה, אל המידע המאוחסן בטבלה C. אם ניחשת שאנו עומדים להשתמש בלולאות FOR-NEXT מוכנסות, אתה מבריק בדיוק כפי שאתה נראה. אך מה בדבר שמות המועמדים? בוא נכניס אותם להוראת DATA ונציב אותם למשתנה המחזרות CS. אם נציב את הוראות CS:PRINT CS בתוך הלולאה הראשונה, ולפני הלולאה השנייה, הן תודפסנה בתחילתה של כל שורה בטבלה. ראה עמוד 00.

50 FOR V=1 TO 3 : READ CS : ? CS,
60 FOR A=1 TO 4 : ? C(V,A), : NEXT A,V
940 DATA POWERMAN, MONEYMAN, OTHER





LIST

5 REM-VOTE COUNTING WITH TWO-DIMENSIONAL ARRAY

```
10 DIM C(3,4)
20 READ V,A : IF A=-9999 THEN 40
30 C(V,A)=C(V,A)+1 : GOTO 20
40 PRINT "CANDIDATE", "18-29", "30-39", "40-49", "50 +" : PRINT
50 FOR V=1 TO 3 : READ CS : PRINT CS,
60 FOR A=1 TO 4 : PRINT C(V,A), : NEXT A,V
900 DATA 1,2, 1,3, 2,2, 2,3, 2,1, 3,2, 3,4, 2,3, 2,3, 3,2, 1,4
910 DATA 3,1, 3,2, 1,4, 2,4, 2,3, 1,3, 2,4, 1,4, 2,2
920 DATA 2,1, 2,2, 3,2, 2,4, 1,2, 1,3, 2,4, 1,4, 2,3, 3,1, 3,3
930 DATA -9999, -9999
940 DATA POWERMAN, MONEYMAN, OTHER
```

OK

RUN

CANDIDATE	18-29	30-39	40-49	50 +
POWERMEN	0	2	3	4
MONEYMAN	2	3	5	4
OTHER	2	4	1	1

OK

קרא



בוא נשיג לעצמנו מידע נוסף הקשור לקובץ הנתונים הזה. בוא נבקש מהמחשב שיתן לנו סך-כול שלוש התשובות לשאלה 1, מבלי להתחשב בקבוצת הגיל.

הואיל ולא השתמשנו במציין אפס במשתנה בעל המציין הכפול $C(V,A)$, "התאים" האלה עדיין נותרו פנויים כדי לאחסן עבורנו מידע. אם כך, נשתמש ב- $C(1,0)$, $C(2,0)$ ו- $C(3,0)$ לחישוב הקולות או התשובות לכל אפשרות בשאלה 1, מבלי להתחשב בקבוצת הגיל. תן דעתך לשימוש במשתנה ה- FOR , V , לצורך הדפסת אחת משלוש האפשרויות (שורה 80).

יכולנו להוסיף עוד הוראת $DATA$ וזה ל-940 ולהשתמש ב- $PRINT CS$:
 $READ CS$ כמו בשורה 50. אנו נותנים בידך את הבחירה.

בצע

הוסף שורות אלו לתכנית הקודמת והרץ את הגרסה החדשה.



70 ? : ? "TOTALS:"

80 FOR V=1 TO 3 : ? "ANSWER" V; " : ";

90 FOR A=1 TO 4 : $C(V,0)=C(V,0)+C(V,A)$: NEXT A

100 ? $C(V,0)$: NEXT V

RUN

CANDIDATE	18-29	30-39	40-49	50 +
POWERMEN	0	2	3	4
MONEYMAN	2	3	5	4
OTHER	2	4	1	1

TOTALS:

ANSWER 1 : 9

ANSWER 2 : 14

ANSWER 3 : 8

OK



ייתכן שיש פיסת מידע נוספת שהיינו רוצים שתדווח לנו: מספרם הכולל של האנשים שהשיבו על השאלון. הרשה לנו איפוא להוסיף עוד קטע קטן לתכנית. שוב אנו מנצלים את העובדה שלא השתמשנו בכל המיקומים בטבלה $C(V,A)$ עם האפס בתור מציין. לחישוב הערכים של $C(V,0)$ ולאחסון מספרם הכולל של המשיבים לשאלוננו, ב- $C(0,0)$, נשתמש בלולאת FOR-NEXT נוספת.

```
110 ? "TOTAL POLLED:";
120 FOR V=1 TO 3 : C(0,0)=C(0,0)+C(V,0) : NEXT V : ? C(0,0)
RUN
```

CANDIDATE	18-29	30-39	40-49	50 +
POVERMAN	0	2	3	4
MONEYMAN	2	3	5	4
OTHER	2	4	1	1

TOTALS:
ANSWER 1 : 9
ANSWER 2 : 14
ANSWER 3 : 8
TOTAL POLLED: 31

OK



קרא

כמובן שיש סוגים אחרים של מידע שנוכל לקבל מנתונים אלה ומתכניתנו זו, בעזרת תוספות אחרות. אתה רשאי לכתוב רוטינה (קטע מתכנית) לחישוב כמה אנשים מכל קבוצת גיל השתתפו בהצבעה. אם אתה בקיא בסטטיסטיקה, תוכל לנסות לכתוב כמה רוטינות לצורך ביצוע הניתוח הסטטיסטי של התכנית, או למציאת אחוזים בקבוצות השונות. השתמש בדימיוןך בתוספת לידע שרכשת בעבודה קשה בשפת ה-BASIC.

מימדים אחרים

מימדים אחרים

אינך חייב לעצור בטבלאות דרמימדיות. למעשה, ה-BASIC בגרסת - Altair מאפשרת את השימוש ב-255 מימדים. יחד עם זאת, בגלל ההגבלה שחלה על אורכה של שורת ההוראה, אינך יכול להכניס הרבה יותר מ-34 מימדים לתוך הוראה אחת.

חלק אותה



להלן שימוש מעניין נוסף בטבלאות. במקרה זה, כשחל ארוע בתכנית, "מסמן" המחשב את התא המתאים בטבלה. תכנית התכנות הבאה היא תכנית "המחלקת קלפים" בצורה אקראית מתוך חבילה של קלפי משחק. "החבילה" הוא למעשה טבלה של 4 על 13 עם תא, המייצג כל אחד מ-13 הקלפים, ב-4 צבעים.

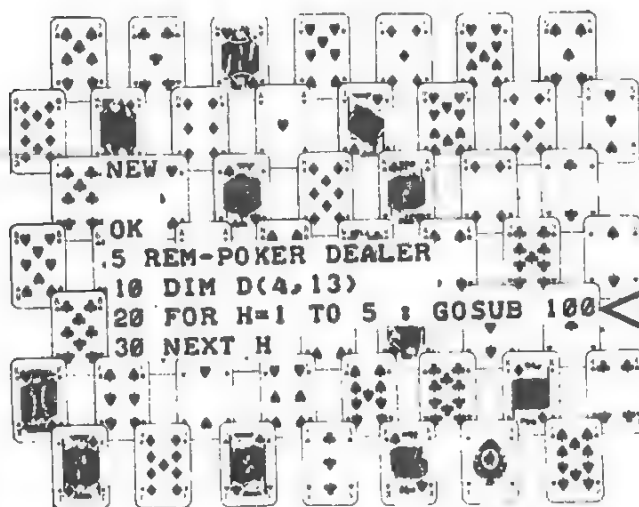
הטבלה $D(4, 13)$ עם כל הערכים הקבועים על אפס, כמו בתחילת ההרצה.

C(1,1) 0	C(1,2) 0	C(1,3) 0	C(1,4) 0	C(1,5) 0	C(1,6) 0	C(1,7) 0	C(1,8) 0	C(1,9) 0	C(1,10) 0	C(1,11) 0	C(1,12) 0	C(1,13) 0
C(2,1) 0	C(2,2) 0	C(2,3) 0	C(2,4) 0	C(2,5) 0	C(2,6) 0	C(2,7) 0	C(2,8) 0	C(2,9) 0	C(2,10) 0	C(2,11) 0	C(2,12) 0	C(2,13) 0
C(3,1) 0	C(3,2) 0	C(3,3) 0	C(3,4) 0	C(3,5) 0	C(3,6) 0	C(3,7) 0	C(3,8) 0	C(3,9) 0	C(3,10) 0	C(3,11) 0	C(3,12) 0	C(3,13) 0
C(4,1) 0	C(4,2) 0	C(4,3) 0	C(4,4) 0	C(4,5) 0	C(4,6) 0	C(4,7) 0	C(4,8) 0	C(4,9) 0	C(4,10) 0	C(4,11) 0	C(4,12) 0	C(4,13) 0

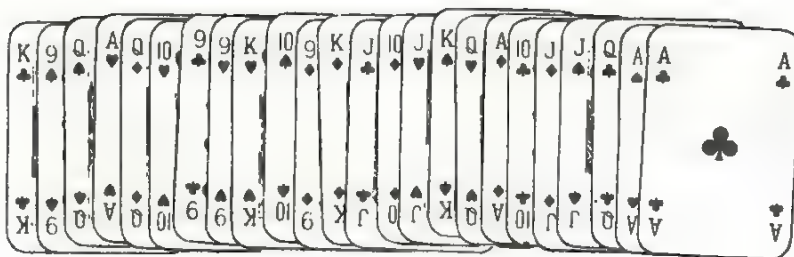
כשהתכנית מורצת, "מסמן" המחשב את התא המתאים בטבלה, ב-1-), כדי להראות שהקלף חולק ואז הוא מדפיס את הקלף והצבע. אנו רוצים שתכניתנו תחלק 5 קלפים, כמו במשחק הפוקר.

המשתנים:

משתנה הפיקוח לשורה FOR-NEXT שמונה עד חמש, בחלוקת חמישה קלפים.



מה זה? ובכן, המשך לקרוא ותבין.



המשתנה המצויין יעקוב אחר הקלפים שיחולקו, עלידי "סימון" בטבלה, כך שהמחשב לא יחלק פעמיים אותו קלף. יש 13 קלפים ב-4 צבעים. שים לב למציינים שבטבלה (13, 4). D. ערכם של המציינים נקבע עלידי שני מספרי RND ובדרך זו ידע המחשב איזה קלף חולק.
J צבעו של הקלף שיש לחלק (ראה שורה 110).
K המספר (1 עד 13) של הקלף שיש לחלק. ראה שורה 100.

99 REM-CARD DEALING SUBROUTINE, LINES 100-330

```
100 K=INT(13*RND(1))+1
110 J=INT(4*RND(1))+1
120 IF D(J,K)=-1 THEN 100
130 D(J,K)=-1
```

בשורה 130 קובעים ערכיהם של J ו-K איזה תא D(J,K) יסומן או ייקבע. יכולנו להשתמש בכל ערך מלבד אפס, כדי לציין שתא מסויים הוחלף מאפס. אך לפני שהתא יסומן ב-1- מבקש המחשב לראות אם "הקלף" חולק כבר פעם, כלומר אם התא שנועד ל-D(J,K) כבר סומן ב-1-. שורה 120 מחזירה את המחשב על עקבותיו, כדי שינסה שנית עם סדרה נוספת של מספרי RND, במקרה שכבר סומן 1- באחד התאים.
בתכנית זו, לכל קלף יש מספר המציין את ערכו, והוא "נשלף" באופן שרירותי מתוך המספרים שבין 1 ל-13, וזאת עלידי שורה 100.

```
100 K=INT(13*RND(1))+1
```

מספר :	1	2	3	4	5	6	7	8	9	10	11	12	13
קלף :	Ace	2	3	4	5	6	7	8	9	10	נסך	מלכה	מלך

כל קלף נושא מספר (1 עד 4), המבטא את הצבע שלו, ונשלף באופן

```
110 J=INT(4*RND(1))+1
```

מספר :	1	2	3	4
צבע :	תלתן	לב שחור	לב אדום	יהלום



שורה 120 מוודאת שהקלף D(J,K) לא חולק עדיין ואם לא, הרי ששורה

130 מסמנת באמצעות 1- שהיא עומדת לחלק קלף, D(J,K).

GOSUB



ההוראה



RETURN



```
5 REM-POKER DEALER
10 DIM D(4,13)
20 FOR H=1 TO 5 : GOSUB 100
30 NEXT H
```



הפעילות העיקרית בתכנית, (שורות 100 עד 330) מבוצעת 5 פעמים, כדי לחלק 5 קלפים. להלן המשך התת-תכנית המחלקת או המדפיסה את הקלפים.

תת תכנית (סוברוטיונה)

```
99 REM-CARD DEALING SUBROUTINE, LINES 100-330
100 K=INT(13*RND(1))+1
110 J=INT(4*RND(1))+1
120 IF D(J,K)=-1 THEN 100
130 D(J,K)=-1
140 ON K GOTO 200,210,210,210,210,210,210,210,210,210,220,230,240
150 ON J GOTO 300,310,320,330
200 ? " ACE " ; : GOTO 150
210 ? K ; : GOTO 150
220 ? " JACK " ; : GOTO 150
230 ? " QUEEN " ; : GOTO 150
240 ? " KING " ; : GOTO 150
300 ? "OF CLUBS" : RETURN
310 ? "OF SPADES" : RETURN
320 ? "OF HEARTS" : RETURN
330 ? "OF DIAMONDS" : RETURN
```



אנו משתמשים בהוראה GOSUB כדי "לקרוא" לתת-תכנית המחלקת קלפים, שורות 100 עד 330, לתוך התכנית. ההוראה GOSUB נמצאת בין ההוראות FOR ו-NEXT (שורות 10 ו-20). התוצאה מכך היא דחיקת חלקה של תת-תכנית זו אל בין ההוראות FOR ו-NEXT. פירוש הדבר בתכניתנו הוא, שתת-תכנית "נקראה" ובוצעה 5 פעמים בעת 5 מסעותיה של הלולאה - FOR-NEXT.

לאחר GOSUB חייב להופיע מספר סידורי (כמו בהוראה GO TO) המורה למחשב היכן מתחילה התת-תכנית. המחשב הולך (GOes) אל התת-תכנית, המתחילה בשורה האמורה, וממשיך להריץ את התכנית, הוראה אחר הוראה, עד שהוא מגיע להוראה RETURN.

כדומה ל-NEXT ו-FOR, ו-READ ו-DATA, גם להוראה GOSUB חייבת תמיד להתלוות ההוראה RETURN. ההוראה RETURN היא תמיד האחרונה בתת-תכנית ומורה למחשב שהוא סיים את התת-תכנית ושעליו לחזור לשורה המופיעה בתכנית מיד אחרי ההוראה GOSUB. על כן חייבות ההוראה GOSUB ו-RETURN להופיע אחרונות בשורה של ריבוי הוראות. התת-תכנית שלנו לחלוקת קלפים עשויה להסתיים במקרה בכל אחת מארבע ההוראות שבשורות 300-330, כך שבמקרה זה יבצע המחשב את ההוראה RETURN אל השורה שאחרי ההוראה GOSUB, לאחר שביצע כל אחת מארבע ההוראות שבשורות 300-330.

עליך לשים לב לכך, שהקפדנו שלא להכניס LINE 30 NEXT H באותה שורה בה מופיעה ההוראה GOSUB. אילו נהגנו כך, הוא היה מחמיץ את ההוראה NEXT, בעת שהיה מבצע את ההוראה RETURN.

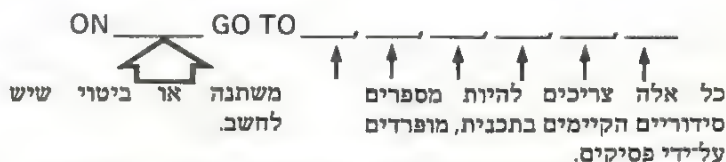
תת תכנית (סוברוטיונה)

ON...GOTO...

ההוראה



בשורות 140 ו-150 אנו מציגים את ההוראה GOTO...ON.



כסך המספרים הסידוריים העשויים להופיע בשורת ההוראה, כן סך המספרים הסידוריים היכולים להופיע אחר ההוראה ON...GOTO. אם המשתנה ON...GOTO הוא 1, אזי עובר המחשב למספר הסידורי הראשון. אם ערך המשתנה הוא 2, הוא עובר למספר הסידורי השני הרשום, וכך הלאה. יחד עם זאת, אם ערכו של משתנה ON...GOTO הוא שלילי, או אפס, או עולה על סך המספרים הסידוריים המופיעים אחרי GO TO, אזי המחשב פשוט מדלג למספר הסידורי הבא בתכנית. הדבר נוח, מפני שאתה עשוי לרצות להשתמש במספרים סידוריים רבים מכפי שאפשר להכליל בשורה. וזהו התכסיס: אמור שרק 10 המספרים הסידוריים בשורה 140 מתאימים לשורה אחת.

140 ON K GOTO 200,210,210,210,210,210,210,210,210,210
התעלמנו משלושת המספרים הסידוריים האחרונים, כשאנו מעמידים פנים שהם לא יתאימו לנו. אנו מורים למשתנה ON...GOTO הבא, להתחיל במקום שהאחרון הפסיק, כך:

145 ON K-10 GOTO 220,230,240



החסר את הערך הגדול ביותר האפשרי של K בפעם האחרונה. K חייב להיות גדול מ-10, כדי לדלג על ההוראה האחרונה ולהגיע לשורה זו.



בדיוק כמו GOSUB ו-RETURN, גם ON...GOTO חייבת להיות את ההוראה האחרונה, אם משתמשים בה בשורה מרובת הוראות, שאם לא כן עליה לעמוד בפני עצמה.

בשורה 140 אנו מוצאים

140 ON K GOTO 200,210,210,210,210,210,210,210,210,220,230,240

שורה זו בוחרת בהוראה שתשמש להדפסת הקלף הנבחר. אם $K = 1$, הולך המחשב לשורה 200 ומדפיס ACE. אם 10 או 9, 8, 7, 6, 5, 4, 3, 2 או $K = 1$ נשלח המחשב לשורה 210 ומדפיס את ערכו של K (מספרו של הקלף שחולק) אם 13 או 12, $K = 11$ הולך המחשב לשורות 220, 230 או 240 ומדפיס JACK, מלכה או מלך.

תן דעתך לכך שההוראה ON...HOTU עשויה לפעמים להיות תחליף לקבוצת הוראות IF...THEN.

ההוראה ON...GOTO בשורה 150 בוחרת את שם הצבע שיש להדפיס. נכון שזכינו לשירות כפול מן הערך RND ל-J ו-K?



ושוב

חלק אותה



עוד שכלול אחד ואז תוכל לנסות את התכנית. הוראות אלו מופיעות אחרי הלולאה FOR...NEXT ואינן מהוות חלק מן התת-תכנית GOSUB 100, לחלוקת קלפים.

```
40 ? : INPUT "ANOTHER HAND, SAME DECK"; AS : IF AS="YES" THEN 20
50 ? : INPUT "ANOTHER HAND, NEW DECK"; AS : IF AS="NO" THEN END
60 FOR J=1 TO 4 : FOR K=1 TO 13 : D(J,K)=0 : NEXT K,J : GOTO 20
```

שורה 60 מתבצעת רק כשהתשובה ל-"ANOTHER HAND, SAME DECK" אינה YES (IF...THEN) ואם התשובה ל-"ANOTHER HAND, NEW DECK" אינה (תנאי שקרי NO(IF...THEN)). שורה 60 משתמשת בלולאות FOR-NEXT מוכנסות כדי לקבוע את כל הערכים שבטבלה D חזרה לאפס.

הערות של הרגע האחרון בנושא GOSUB.

ייתכן שיהיו לך תת-תכניות בתוך תת-תכנית, כלומר GOSUB בתוך GOSUB, בדומה ללולאות FOR-NEXT מוכנסות. הוראת ה-RETURN בה נתקל המחשב, מחזירה את התכנית אל השורה שאחרי השורה בה בוצעה הוראת ה-GOSUB האחרונה. הגיוני, נכון? בנוסף לכך עלינו גם להודיע שההוראה ON...GOTO היא דו-דנית קרובה להוראה ON...GOSUB. יכולנו להשתמש בה בתכנית חלוקת הקלפים כתחליף לשורות הבאות:

```
140 ON K GOSUB 200,210,210,210,210,210,210,210,210,210,220,230,240
200 ? " ACE " ; : RETURN
210 ? K ; : RETURN
220 ? " JACK " ; : RETURN
230 ? " QUEEN " ; : RETURN
240 ? " KING " ; : RETURN
```

תכנית זו דומה לתת-תכנית בתוך תת-תכנית, פרט לעובדה שההוראה ON...GOSUB הוכנסה "לתוך" התת-תכנית. אין תשובות למספר השורה אליו מופנה המחשב עלידי הוראת ONK GOSUB והוא יחזור לשורה 150 (השורה שאחרי ה-GOSUB האחרון) הואיל והצבנו הוראת RETURN בסופן של כל השורות שאליו יכול היה ה-GOSUB לפנות.

כעת, כשהתכנית ברורה כשמש (מוטב שתחזור ותעיין בה ואף תבדוק את ההסברים שלצדה, אם אינך חש כך) קדימה הרץ אותה.

LIST

5 REM-POKER DEALER

10 DIM D(4,13)

20 FOR H=1 TO 5 : GOSUB 100

30 NEXT H

40 PRINT : INPUT "ANOTHER HAND, SAME DECK"; AS : IF AS="YES" THEN 20

50 PRINT : INPUT "ANOTHER HAND, NEW DECK"; AS : IF AS="NO" THEN END

60 FOR J=1 TO 4 : FOR K=1 TO 13 : D(J,K)=0 : NEXT K,J : GOTO 20

99 REM-CARD DEALING SUBROUTINE, LINES 100-330

100 K=INT(13*RND(1))+1

110 J=INT(4*RND(1))+1

120 IF D(J,K)=-1 THEN 100

130 D(J,K)=-1

140 ON K GOTO 200,210,210,210,210,210,210,210,210,210,220,230,240

150 ON J GOTO 300,310,320,330

200 PRINT " ACE "; : GOTO 150

210 PRINT K; : GOTO 150

220 PRINT " JACK "; : GOTO 150

230 PRINT " QUEEN "; : GOTO 150

240 PRINT " KING "; : GOTO 150

300 PRINT "OF CLUBS" : RETURN

310 PRINT "OF SPADES" : RETURN

320 PRINT "OF HEARTS" : RETURN

330 PRINT "OF DIAMONDS" : RETURN

OK

RUN

QUEEN OF HEARTS

8 OF HEARTS

KING OF DIAMONDS

10 OF HEARTS

6 OF SPADES

ANOTHER HAND, SAME DECK? YES

3 OF HEARTS

4 OF CLUBS

JACK OF SPADES

9 OF HEARTS

2 OF DIAMONDS

ANOTHER HAND, SAME DECK? NO

ANOTHER HAND, NEW DECK? YES

JACK OF DIAMONDS

8 OF SPADES

JACK OF CLUBS

7 OF CLUBS

QUEEN OF DIAMONDS

ANOTHER HAND, SAME DECK? YES

QUEEN OF HEARTS

6 OF SPADES

3 OF CLUBS

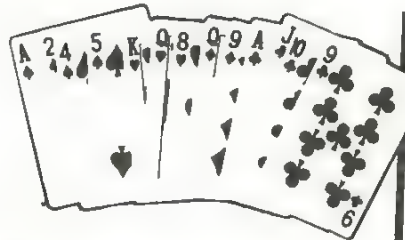
4 OF SPADES

7 OF DIAMONDS

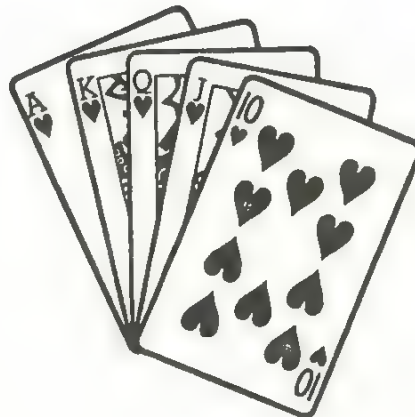
ANOTHER HAND, SAME DECK? NO

ANOTHER HAND, NEW DECK? NO

OK



מיוחד למומחים: הרחב את התכנית כך שהמחשב יוכל לעקוב אחר מספר הקלפים שחולקו ולקבוע את ערכי הטבלה אוטומטית לאפס, אחרי שחולקו 52 קלפים ולפני שהמחשב מנסה לחלק את הקלף ה-53. בתכנית זו, יעבור המחשב בלולאה משורה 120 לשורה 100, כשהוא מחפש מקום נטול 1- בטבלה, לאחר שהספיק לחלק 52 קלפים. מה דעתך על תכנית שתחלק קלפים במשחק העשרים ואחד?



ההוראה GOSUB/RETURN

גורמת לתכנית לקפץ או להסתעף אל הוראה אחרת ולהמשיך עד שהיא נתקלת ב-RETURN, נקודה בה חוזרת התכנית אל ההוראה המופיעה אחרי GOSUB.

```
10 GOSUB 200
20 PRINT "END OF SUBROUTINE DEMO"
30 END (Use STOP or END to separate program from subroutines)
200 REM – SUBROUTINE
210 PRINT "THIS IS A SUBROUTINE"
220 RETURN
```

ההוראה DIM

מקציבה מקום לטבלאות, מטריצות או מטריצות מחרוזות וקובעת את כל ערכי המטריצה לאפס. כל המצינים מתחילים באפס כשאין אפס.
to be 11 elements (1 – 10).

DIM (משתנה) (N) (משתנה) (N,M) ...

```
10 DIM A(3), B(5;10)
20 DIM N$(10,10)
```

Dynamic DIMension

```
10 N = 30
20 DIM X(N*30)
```

BASIC PLUS only

רק בגירסת BASIC PLUS

מציין את מספר המחרוזות ולא את ארכה של כל אחת מהן. DIM A\$(3)

ההוראה ON-GOTO

מקפצת או מדלגת (GOSUB) אל ההוראה שמספרה מצויין עלידי המספר N, שאחרי GOTO.

מספר סידורי, מספר סידורי, מספר סידורי; GOTO (משתנה) NO מספר סידורי.

```
10 ON GOTO 100, 200, 300
```

אם $N = 1$ קופצים לשורה 100

אם $N = 2$ קופצים לשורה 200

וכו'

```
20 ON N GOSUB 100, 200, 300
```

כמו דלעיל, מלבד העובדה ש-RETDRN מ-GOSUB מדלג חזרה אל ההוראה הבאה אחרי ON GOSUB.

1. מתי הינך משתמש בהוראה DIM בטבלה דרממדיית?
2. אתה עובד במחלקת כוח האדם בכפר נופש גדול, כדוגמת דיסנילנד. הממונה עליך יוצא מגדרו כדי להוכיח לך שאתה זוכה להזדמנות נאותה להפגין את יכולתך. הוא מבקש ממך להכין תכנית מחשב שבעזרתה ניתן יהיה להגיע לחלוקת כל העובדים, לפי גיל ומין. הוא מספק לך מידע כדלהלן, (ועליך מוטל לסדרו בזוגות, בהוראות DATA).

זכר = 1 נקבה = 2

גיל

מתחת ל-21 = 1

21 - 29 = 2

30 - 39 = 3

40 - 49 = 4

50 - 59 = 5

over 60 = 6

REPORT

AGE

MALE

FEMALE

TOTAL

UNDER 21

21 - 29

30 - 39

40 - 49

50 - 59

OVER 60

3. דוח מכירות 2: זוכר את דו"ח המכירות של Natiowide Peddlars? הוא היה חסר רק נתון אחד, כדי להיות מושלם באמת; בכל אחד מששת אזורי המכירה פעלו שלושה אנשי מכירות, שכל אחד מהם מדווחים על מכירותיהם למשרדה הראשי של החברה; הדו"חות מועברים בטלפון, כשהאזור מצויין על ידי מספר, איש המכירות מצויין במספר והמכירות בדולרים. כתוב תכנית שתאגד בתוכה את המכירות והדפס דו"ח נאה כמו בהמשך.

סה"כ	איש מכירות 3	איש מכירות 2	איש מכירות 1	אזור
1				
2				
3				
4				
5				
6				

4. מדריך לקניות: בחר 10 מוצרים בשמם ובמידותיהם. בחר ארבע חנויות, ערוך בהן סיור ורשום את מחירו של כל פריט. כתוב תכנית מחשב שתראה כל חנות, את המחירים שלה ואת העלות הכוללת אילו קנית את כולם בחנות אחת ויחידה. המומחים יכולים להוסיף * ליד המחיר הזול ביותר לכל פריט. סדר את המידע בהוראות DATA, עבור כל חנות בנפרד.

פונקציות

RND(X) - יוצרת מספר אקראי שבין 0 ל-1.
 ל - RND ב - BASIC בגירסת Altair, ראה עמודים 74-75.

הפונקציות הבאות מתאימות הן ל-BASIC PLUS והן ל-ALTAIR BASIC.
 LEN(\$) - נותנת את המספר השלם השווה למספר התווים במשתנה.
 DEF FNA (A) - הגדר את הפונקציה שלך.
 ABS(X) - נותנת ערך מוחלט לביטוי X.
 INT(X) - נותנת את השלם הגדול ביותר, הקטן מהארגומנט X או השווה לו.
 TAT(X) - יוצרת רווחים וטורים בהדפסה על המסוף.
 USR(X) - קוראת לתת-תכנית X.
 FRE(O) - נותנת את מספר הסיבית (bytes) שלא הופיעו בזיכרון.
 SPC(X) - מדפיסה X שורות ריקות.
 SGN(X) - נותנת 1 אם X גדול מ-0, 0 אם X הוא 0, ו-1 אם X קטן מאפס.
 SIN(X) - נותנת את סינוס הביטוי X אם X מופיע ברדיאנים.
 SQR(X) - נותנת את השורש הרבועי של y.
 ATN(X) - נותנת את הארקטנגנס של X ברדיאנים.
 COS(X) - נותנת את הקוסינוס של X ברדיאנים.
 POS(X) - נותנת את המיקום העכשוי של ההדפסה.

DEF FN מאפשרת להגדיר פונקציות.
 פונקציה = (משתנה מדומה) (שם משתנה) DEF FN מספר סידורי.

רק ל-BASIC PLUS
 FIX(X) - מחזירה את ערכו המקוצץ של X.
 LOG*(X) - נותנת את הלוג הנפוץ של X.
 PI - הערך הקונסטנטי של 3.1415927.
 RND - יוצרת מספר אקראי בין 0 ל-1. אותו סדר בכל הרצה. השתמש בהוראה
 RANDOMIZE 10 לשינוי הסדר.

פונקציות מחרוזת

רק ל-BASIC PLUS
 מחרוזות (A\$,B\$,N!) מחפשת מחרוזת A\$ לתת-מחרוזת B\$. כשהיא מתחילה
 בתו NI במחרוזת A\$. נותנת אפס אם אינה מוצאת את התת-מחרוזת. נותנת את
 מיקום התווים אם התת-מחרוזת אכן נמצאת.
 רווחים (N%) - מכינסה N רווחים בתוך מחרוזת.
 פונקציות המחרוזת ל-ALTAIR BASIC, ראה עמודים 98-99.

יופי, הגעתם לסוף הספר, אך עדיין לא הגעתם לסופה של ה-BASIC, ייתכן שבמערכת המחשבים שלכם משתמשים בגירסת BASIC, בעלת אפשרויות רבות יותר מאשר כוסו עלידינו, ואנחנו אפילו לא הספקנו לכסות את ה-BASIC בגירסת ALT&K. יחד עם זאת, אם הינכם מבינים ומסוגלים להפעיל את ההוראות והפונקציות שהצגנו בפניכם, עליכם על דרך המלך. אם הינכם מסוגלים לפתור את הבעיות בסופו של כל פרק, הינכם מתקדמים ליכולת לכתוב תכניות לפי צרכיכם.

השתעשעו והמשיכו לתרגל.

המחבר ישמח לקבל את הערותיכם, הצעותיכם ודברי הביקורת שלכם. האם מצאתם טעויות? מה בילבל אתכם? (האם יש שיטות טובות יותר להסביר את הדברים? דוגמאות מוצלחות יותר? נמשיך לשפר ספר זה לפני הדפסת כל מהדורה נוספת, כך שהערותיכם אכן תזכינה לתשומת לבו של המחבר, המבטיח להשיב על מכתביכם.

אפריל 1977

תודה

ג'ראלד ריבאראון

דיימקס

ת.ד. 410

מנלו פארק, קליפורניה 94025



המחבר הוא בעל ב.א. בפסיכולוגיה ומ.א. במחקר וחינוך, מאוניברסיטת הארוורד. הוא סגן נשיא חברת "דיימקס", והשתתף בהקמתה של "People Computer Company" וחברת הבת שלה במנלופארק, קליפורניה. יחד עם אלכרכט, פינקל ובראון חבר את הספר "BASIC" (שיצא לאור ב-1973 בהוצאת וילי). כמובן הוא בימאי סרטים והפיק סרטים רבים לטלוויזיה הלימודית.



פרק 1 פתרונות

1.

```
10 PRINT "LISA STEWART"
20 PRINT"1826 GOLDEN AVE"
30 PRINT"REDWOOD CITY, CA"

RUN
LISA STEWART
1826 GOLDEN AVE
REDWOOD CITY, CA
```
2.

```
10 PRINT"LISA STEWART","1826 GOLDEN AVE","REDWOOD CITY,CA"
OK
RUN
LISA STEWART  1826 GOLDEN AVE          REDWOOD CITY,CA
```
3.

```
NEW
RETURN
OK
```
4.

```
LIST
RETURN
```
5.

```
SHIFT  ←   (back arrow)
or underline.
```
6.

```
OK
PRINT 172.16-13.50-19.00-3.25-10.00-114.14+87.51
99.78
```
7.

```
10 LET T=(40+50+46+48+49+45+52+41+44+46)/10
20 PRINT "AVERAGE HEIGHT IS",T

RUN
AVERAGE HEIGHT IS          46.1
```

פרק 2 פתרונות

1.

```
LET
INPUT
READ
```
2. מרכאות
3. השתמש בגישה ישירה
4.

A	B	C	D
12	0	0	0
12	14	0	0
12	14	16	0
12	14	16	0
12	14	16	5
12	14	16	5
5.

```
10 READ A$,B$,A
20 PRINT A$,B$,A
30 DATA "SHERRY DELIGHT","800-555-1212",23

RUN
SHERRY DELIGHT          800-555-1212  23
```
6.

```
10 INPUT"ENTER YOUR NAME"JA$
20 INPUT "ENTER YOUR ASTROLOGICAL SIGN"JS$
30 INPUT "ENTER YOUR BIRTHDATE"JD$
40 PRINT A$,"YOU ARE UNDER THE SIGN OF "JS$
50 PRINT"SINCE YOUR BIRTHDAY IS "JD$

RUN
ENTER YOUR NAME? JERRY
ENTER YOUR ASTROLOGICAL SIGN? LEO
ENTER YOUR BIRTHDATE? 8/13/1950
JERRY          YOU ARE UNDER THE SIGN OF LEO
SINCE YOUR BIRTHDAY IS 8/13/1950
```



```

7. 10 LET X=((4.80/.48)*100)*7.5)/150
    20 PRINT X,"MONTHS"
    OK
    RUN
    50 MONTHS

8. 20 INPUT"ENTER NO. OF SHOWERS PER DAY";S
    30 INPUT "ENTER NO. OF MINS. PER SHOWER";M
    40 INPUT"ENTER NO. OF TUB BATHS PER DAY";B
    50 INPUT"ENTER NO. OF HAND DISHWASHING JOBS/DAY";D
    60 INPUT"ENTER NO. OF AUTO. DISHWASHING JOBS/DAY";A
    70 INPUT"NO. OF TOILET FLUSHES/DAY";F
    80 INPUT"NO. OF WASHER LOADS/WEEK";W
    90 INPUT"NO. OF OUTDOOR HOSE MINUTES/DAY";H
    100 INPUT"ENTER NO. OF VARIOUS GALLONS USED/DAY";V
    110 LET T=0
    120 READ X
    130 LET T=T+(S*(M*X))*30
    140 READ X
    150 LET T=T+(B*X)*30
    160 READ X
    170 LET T=T+(D*X)*30
    180 READ X
    190 LET T=T+(A*X)*30
    200 READ X
    210 LET T=T+(F*X)*30
    220 READ X
    230 LET T=T+(W*X)*4.2
    240 READ X
    250 LET T=T+(H*X)*30
    260 LET T=T+(V*30)
    270 PRINT"YOU USE APPROX. "; T;"GALLONS / MONTH OR ";T/7.5;" CUBIC FEET
    280 PRINT"THAT IS AN AVERAGE OF ";T/30;"GALLONS PER DAY"
    300 DATA 6,20,15,16,6,35,10

RUN
ENTER NO. OF SHOWERS PER DAY? 1
ENTER NO. OF MINS. PER SHOWER? 5
ENTER NO. OF TUB BATHS PER DAY? 1
ENTER NO. OF HAND DISHWASHING JOBS/DAY? 1
ENTER NO. OF AUTO. DISHWASHING JOBS/DAY? 1
NO. OF TOILET FLUSHES/DAY? 5
NO. OF WASHER LOADS/WEEK? 3
NO. OF OUTDOOR HOSE MINUTES/DAY? 0
ENTER NO. OF VARIOUS GALLONS USED/DAY? 50
YOU USE APPROX. 5271 GALLONS / MONTH OR 702.8 CUBIC FEET
THAT IS AN AVERAGE OF 175.7 GALLONS PER DAY

```

פרק 3 פתרונות

1. 896700
1,000,000,000,000,000
1001
61,576,200,000
.00000387124
2. 1.78643E6
3.1457E6
1.2479E-4
4.2456E-1
3. RETURN לחץ על
4. CONTROL/C

```

5. 10 PRINT "NUMBER","SQUARED"
    20 LET T=1
    30 PRINT T,T*2
    40 LET T=T+1
    50 GOTO 30

```

NUMBER	SQUARED
1	1
2	4
3	9
4	16
5	25
6	36
7	49
8	64
9	81
10	100

```

6. 20 PRINT "F","C"
    30 LET F=30
    40 PRINT F,(5/9)*(F-32)
    50 LET F=F+1
    60 GOTO 40

```

RUN F	C
30	-1.11111
31	-.555556
32	0
33	.555556
34	1.11111
35	1.66667
36	2.22222
37	2.77778
38	3.33333
39	3.88889
40	4.44445
41	5
42	5.55556

פרק 4 פתרונות

אין פתרונות

פרק 5 פתרונות CHAPTER 5 SOLUTIONS

- ממשיך להוראה הבאה הקרובה ביותר בתכנית.
 - לא שווה
 - פחות מ- או שווה ל-.
 - אין הוראת print והיא מדלגת לאותו מקום (שורה, 30) אם התנאי נכון או לא נכון.
- ```

5. 10 INPUT "ENTER WATER USED IN GALLONS?" G
 20 LET T=(G/7.5)/100:LET T=(T*.50)+2.85:PRINT"BILL IS "T:GOTO 10

```
- RUN
- ```

ENTER WATER USED IN GALLONS? 6000
BILL IS 6.85
ENTER WATER USED IN GALLONS? 10000
BILL IS 9.51667
ENTER WATER USED IN GALLONS? 20000
BILL IS 16.1833
ENTER WATER USED IN GALLONS?

```

```

6. 10 INPUT "ENTER WATER USED IN GALLONS";G
    20 IF G<=8000 THEN PRINT "BILL IS "((G/7.5)/100)*.50+2.85;GOTO 10
    30 LET T=((G-8000)/7.5)/100*.100
    40 LET T=T+((8000/7.5)/100)*.50+2.85;PRINT"BILL IS";T;GOTO 10

RUN
ENTER WATER USED IN GALLONS? 6000
BILL IS 6.85
ENTER WATER USED IN GALLONS? 10000
BILL IS 10.85
ENTER WATER USED IN GALLONS? 20000
BILL IS 24.1833
ENTER WATER USED

7. 10 INPUT "HOW MANY MINUTES";M
    20 LET C=2.00;IF M<=45 THEN LET C=C+(M*.05);GOTO 40
    30 LET C=C+(45*.05)+(M-45)*.03
    40 PRINT"BILL IS ";C; GOTO 10

RUN
HOW MANY MINUTES? 40
BILL IS 4
HOW MANY MINUTES? 45
BILL IS 4.25
HOW MANY MINUTES? 60
BILL IS 4.7
HOW MANY MINUTES? 100
BILL IS 5.9
HOW MANY MINUTES? 0DAD NS,ZS
20 IF Z$="94061"THEN PRINTNS;GOTO 10
30 GOTO 10
40 DATA MARCUS,94025,LINDA, 94061, JERRY, 94061,LARRY,06542
OK
RUN
LINDA
JERRY

```

פרק 6 פתרונות

```

1 ערך שלילי ל- x
2 3, -4, 4, -5, 0, -1
3. 10 FOR X = 1 TO 30: ? INT(6*RND(1)) + 1 : NEXT
4. אין פתרון
5. 10 REM CRAPS
    20 LET A=INT(6*RND(1)+1);LET B=INT(6*RND(1)+1)
    25 PRINT"POINT IS "A+B
    30 IF A+B=7 THEN PRINT"WINNER";GOTO 20
    40 IF A+B=11 THEN PRINT"WINNER";GOTO 20
    50 LET C=INT(6*RND(1)+1);LET D=INT(6*RND(1)+1)
    55 PRINT C+D,
    60 IF A+B=C+DTHEN PRINT"WINNER";GOTO 20
    70 IF C+D=7 THEN PRINT"YOU CRAPPED OUT";GOTO 20
    80 GOTO 50

RUN
POINT IS 6
4 5 10 4 8
8 9 9 6 WINNER
POINT IS 11
WINNER
POINT IS 6
7 YOU CRAPPED OUT
POINT IS 5

```

```

6. 10 REM STARS
    20 LET N=INT(100*RND(1)+1)
    30 INPUT"ENTER YOUR GUESS";G:IF G=N THEN PRINT"WINNER":GOTO 20
    40 LET D=ABS(G-N)
    50 IF D>=64 THEN 170
    60 IF D>=32 THEN 160
    70 IF D>=16 THEN 150
    80 IF D>=8 THEN 140
    90 IF D>=4 THEN 130
    100 IF D>=2 THEN 120
    110 PRINT"*";
    120 PRINT"*";
    130 PRINT"*";
    140 PRINT"*";
    150 PRINT"*";
    160 PRINT"*";
    170 PRINT"*":GOTO 30

```

```

RUN
ENTER YOUR GUESS? 50
**
ENTER YOUR GUESS? 25
***
ENTER YOUR GUESS? 15
*****
ENTER YOUR GUESS? 10
*****
ENTER YOUR GUESS? 8
*****
ENTER YOUR GUESS? 12
*****
ENTER YOUR GUESS? 13
*****
ENTER YOUR GUESS? 11
WINNER
ENTER YOUR GUESS

```

פרק 7 פתרונות

1. הוא מדפיס 17 ונעצר
2. ההדפסה תמשך בכל פעם שתועבר בלולאה. $RND(0)$ אותו מספר בכל פעם.
3. אין פתרון.
4. 10 FOR X = 1 TO 72 : PRINT "": NEXT
5.

```

5 REM INTEREST TABLE
6 PRINT"YEARS","5%", "5.5%", "6%", "6.5%"
10 FOR Y=5 TO 25 STEP 5
15 PRINT Y,
20 FOR I=5 TO 6.5 STEP .5
30 PRINT 10000*(1+(I/100))^Y,:NEXT I:PRINT
40 NEXT Y

```

YEARS	5%	5.5%	6%	6.5%
5	12762.8	13069.6	13382.3	13700.9
10	16289	17081.4	17908.5	18771.4
15	20789.3	22324.7	23965.6	25718.4
20	26533	29177.5	32071.3	35236.5
25	33863.6	38133.7	42918.7	48277.1


```

6. 10 REM PAYROLL
    15 PRINT"NAME","GROSS PAY"
    20 READ N$,H,R
    30 IF H<=40 THEN PRINTN$,H*R:GOTO 20
    40 LET G=(40*R)+(H-40)*(R*1.5):PRINTN$,G:GOTO 20
    900 DATA T. BOD, 40,4.00, T. RAY, 50, 4.00

RUN
NAME           GROSS PAY
T. BOD          160
T. RAY          220

700 ERROR IN 20
OK

```

פרק 8 פתרונות

```

1. 10 DEF FNR(X)=INT(X+100+.5)/100

2. BASIC IS BEST

3. 20 AS = "INSTANT BASIC"
    30 FOR X = LEN(AS) TO 1 STEP -1
    40 PRINT MID(AS,X,X),
    50 NEXT X

4. 10 READ N$
    20 FOR X=1 TO LEN(N$)
    30 IF MID$(N$,X,1)<>"",THEN 60
    40 PRINT MID$(N$,X+1),LEFT$(N$,X-1)
    60 NEXT X
    80 DATA"ZENITH,HAROLD"

5. אין פתרון.

6. 5 REM RANDOM NAME GENERATOR(CVCCVC)
    10 V$="AEIOU":N$="BCDFGHJKLMNPQRSTVWXYZ"
    15 FOR N=1 TO 20
    20 V1=INT(5*RND(1))+1:V2=INT(5*RND(1))+1
    30 C2=INT(21*RND(1))+1:C3=INT(21*RND(1))+1:C4=INT(21*RND(1))+1
    40 PRINT"J";MID$(V$,V1,1);MID$(N$,C2,1);MID$(N$,C3,1);
    45 PRINT MID$(V$,V2,1);MID$(N$,C4,1),
    50 NEXT N

RUN
JIPFID          JUDBUV          JIDRID          JUHWEK          JEPTIZ
JOJWEB          JAXJOK          JEPSIG          JEZSUK          JIGNAV
JUSPOH          JEPXAT          JIYBOT          JUHLIF          JEMJEK

```

פרק 9 פתרונות

- שורה 70
הדפס שמו של איש המכירות ואת סך כל המכירות
שורות 100 או 110
- L. FRENCH
B. MIDLER
1400
500
- 16
9
9
5
- רק אם הטבלה שלך מכילה יותר מ-11 משתנים.

פרק 10 פתרונות

1. תמיד

```

2. 10 REM EQUAL OPPORTUNITY
15 DIM C(6,2)
20 READ S,A:IF S=-1 THEN 43
30 C(A,S)=C(A,S)+1:C(0,S)=C(0,S)+1:C(A,0)=C(A,0)+1:GOTO 20
40 PRINT"AGE","MALE","FEMALE","TOTAL"
50 FOR A=1 TO 6:READ D$:PRINTD$,
60 FOR S=1 TO 2:PRINT C(A,S),:NEXT S
70 PRINTC(A,0):NEXT A
80 PRINT"TOTALS",C(0,1),C(0,2)
90 DATA 1,3,2,2,1,1,2,4,2,5,2,6,1,6,1,5,1,4,1,3,1,2,1,2,-1,-1
100 DATA UNDER 21,21-29,30-39,40-49,50-59,OVER 59

```

AGE	MALE	FEMALE	TOTAL
UNDER 21	1	0	1
21-29	2	1	3
30-39	2	0	2
40-49	1	1	2
50-59	1	1	2
OVER 59	1	1	2
TOTALS	8	4	

```

3. 10 REM SALES REPORT 2
20 DIM G(6,3)
30 READ T,S,D:IF T=-1 THEN 50
40 G(T,S)=G(T,S)+D:G(T,0)=G(T,0)+D:GOTO 30
50 PRINT"TERRITORY","SALNO1","SALNO2","SALNO3","TOTAL"
60 FOR T=1 TO 6:PRINTT,
70 FOR S=1 TO 3:PRINTG(T,S),:NEXT S:PRINTG(T,0):NEXT T
90 DATA 1,1,500,2,1,1000,3,1,200,2,1,150,2,1,400,2,3,500
100 DATA 3,1,1000,3,2,2000,3,3,1500,4,1,1500,4,2,1500,4,3,2000
110 DATA 5,1,100,5,2,100,5,3,100
120 DATA 6,1,650,6,2,760,6,3,800
130 DATA -1,-1,-1

```

TERRITORY	SALNO1	SALNO2	SALNO3	TOTAL
1	500	0	0	500
2	1550	0	500	2050
3	1200	2000	1500	4700
4	1500	1500	2000	5000
5	100	100	100	300
6	650	760	800	2210

```

4. 10 REM SHOPPING GUIDE
15 DIM P(10,4),T(4)
18 REM READ IN DATA
20 FOR S=1 TO 4
30 FOR I=1 TO 10: READ P(I,S):NEXT I:NEXT S
40 REM PRINT IT OUT
50 PRINT"ITEM #","STORE1",2,3,4
60 FOR I=1 TO 10:PRINTI,
70 FOR S=1 TO 4:PRINTP(I,S),:T(S)=T(S)+P(I,S):NEXT S
80 NEXT I
90 PRINT"TOTALS",:FOR X=1 TO 4:PRINTT(X),:NEXT X
100 DATA 1,2,3,4,5,6,7,8,9,10
110 DATA 10,9,8,7,6,5,4,4,2,1
120 DATA 1,1,2,2,3,3,4,4,5,5
130 DATA4,4,5,5,6,6,7,7,8,8

```

ITEM #	STORE1	2	3	4
1	1	10	1	4
2	2	9	1	4
3	3	8	2	5
4	4	7	2	5
5	5	6	3	6
6	6	5	3	6
7	7	4	4	7
8	8	4	4	7
9	9	2	5	8
10	10	1	5	8
TOTALS	55	56	30	60

```

5. 20 READ P1,V1,T1:IF P1=-1 THEN 50
    30 P(P1)=P(P1)+1:V(V1)=V(V1)+1:T(T1)=T(T1)+1:GOTO 20
    50 PRINT"PRESIDENT","VEEP","TREASURER"
    60 FOR X=1 TO 3
    70 PRINTX:P(X),X:V(X),X:T(X)
    80 NEXT X
    90 DATA 1,1,1,2,2,2,3,3,3,1,2,3,3,2,1,2,1,3,2,2,3,1,2,1,-1,-1,-1

```

```

RUN
PRESIDENT      VEEP      TREASURER
1 3            1 2        1 3
2 3            2 5        2 1
3 2            3 1        3 4

```

```

6 10 REM GRADE COUNTER
    15 PRINT"GRADE","NUMBER"
    20 READ G:IF G=-1 THEN 30
    25 C(G)=C(G)+1:GOTO 20
    30 FOR X=1 TO 5:PRINTX,C(X):NEXT X
    40 PRINT "A'S AND B'S TOTAL",C(4)+C(3)
    50 DATA 4,4,3,3,2,2,1,1,3,3,4,4,4,3,2,3,4,-1

```

```

RUN
GRADE          NUMBER
1              2
2              3
3              6
4              6
5              0
A'S AND B'S TOTAL      12

```

```

7. 5 REM SLOT MACHINE
    10 FOR X=1 TO 5:READ N$(X):NEXT X
    20 FOR X=1 TO 3:A(X)=INT(5*RND(1))+1:NEXT X
    30 FOR X=1 TO 3:PRINT N$(A(X)),:NEXT X
    55 REM DOUBLE /TRIPLE TEST
    60 IF A(1)<>A(2) THEN 80
    65 IF A(2)<>A(3) THEN 70
    66 IF A(1)=5 THEN PRINT"BIG WINNER. TAKE $1.00":GOTO 20
    67 PRINT"3 IN A ROW. $.25":GOTO 20
    70 PRINT"YOU WIN $.10":GOTO 20
    75 REM CHERRY TEST
    80 IF A(1)=1 THEN PRINT"YOU WIN $.05":GOTO 20
    90 PRINT"NOTHING WON. NEXT NICKEL":GOTO 20
    100 DATA CHERRY,PEACH,PLUM,APPLE,BAR

```

```

RUN
PLUM           PEACH           APPLE           NOTHING WON. NEXT NICKEL
CHERRY         BAR            APPLE           YOU WIN $.05
APPLE          PLUM           PEACH         NOTHING WON. NEXT NICKEL
BAR            PLUM           PLUM          NOTHING WON. NEXT NICKEL
CHERRY         BAR            PLUM          YOU WIN $.05
PEACH          BAR            APPLE         NOTHING WON. NEXT NICKEL
APPLE          APPLE         BAR            YOU WIN $.10
BAR            APPLE         BAR

```

שעשועי מחשב כיס

ג'ימס ט. רוג'רס

מה תוכל לעשות במחשב הכיס שלך אחרי שבדקת את חשבון ההמחאות שלך, חישבת את ההוצאות החודשיות של משק הבית או הכנת שיעורי בית במתמטיקה? "שעשועי מחשב כיס" מהווה תשובה קולעת לשאלה זו. זהו קובץ נהדר של משחקים, חידות, שעשועי לשון, תכסיסים, מעשי קסמים מתמטיים ושעשועים אחרים שניתן לבצע בעזרת ההמצאה הגדולה של שנות השבעים, מחשב הכיס.

היכן תוכל לקנות גאו בתקופת משבר האנרגיה?

הכפל 142.15469 ב-5 וקרא את התשובה מלמעלה למטה.

או, הכפל 284.02212 ב-2.5.

האם תרצה לברך חבר בדרך אלקטרונית? נסה להפוך 107.734.

נחש את גילו של חברך, את יום הולדתו, היזכר בתאריכים חשובים בדברי ימיה של אמריקה. תוכל לעשות כל זאת, ואף יותר בעזרת המכשיר הספר "שעשועי מחשב הכיס".

מחברם של השעשועים והחידות הוא ג'ימס ט. רוג'רס. כבוגר אוניברסיטת הרווארד, עבד שנים רבות כעיתונאי, וכיום הוא מכהן כחבר במועצת העורכים של ה"סיינטיפיק אמריקן". הוא עובד בניו-יורק ומתגורר במונטרוז, פנסילבניה ובניו-יורק.

הספר — הראשון — והיחיד — מסוגו — אשר
ילמדך ליהנות ממחשב הכיס שלך

משחקי מחשב כיס

מאת ד"ר אדווין שלוטברג וג'ון ברוקמן

מעכשו תוכל לנצל את מחשב
הכיס שלך ככלי משחק, כתשבץ,
כחפיסת קלפים וכחידת מבוכים.
בצורה זו הופך מחשב-הכיס לכלי
חברתי המשעשע, המבדר ומקשר
בין אנשים.

לפניך 50 משחקים וחידות אשר אתה
יכול לשחק ולפתור בעצמך עם
מתחרה אחר או יותר.

זהו הספר הראשון המלמדך תוך
כדי הנאה כיצד להפעיל את מוחך
כאשר אתה משתמש במחשב הכיס.

ספר החוקים
למשחקים חשבוניים חרישיים
שהנך יכול לשחק
עם מחשב הכיס האלקטרוני שלך.

המחשב

כלי ההווה והעתיד

קלאוד דה רוסי

ספרנו הוא מבוא לידע המחשב. הוא נועד לתת לקורא מושג על המחשב, ובאיזה אופן אפשר להעבירו. התמונות הכלולות בו באות לעזור בהסברת המונחים המתארים את חלקי המחשב ואת השיטות הננקטות לשם מיחשוב ופתרון בעיות שונות.

בפתח הספר — היסטוריה קצרה של התפתחות תעשיית המחשבים. מכיוון שבעתיד אנו עשויים לחיות בעולם המופעל כולו על ידי מחשבים, מקווה אני שתוכן הספר יתרום להיכרות בין הקורא ובין "אבני היסוד" של עולם העתיד.

על המחבר

עבודתו בת שמונה השנים בשטח המחשבים של קלוד י. דה רוסי, יחד עם התענינותו העמוקה בנוער, הביאה אותו לידי כתיבת ספר זה בו הוא מציג לפני הנוער את עולם המחשבים. המחבר דה רוסי, מתגורר עם אשתו ושני ילדיו הצעירים באמסטרדם, ניו יורק. הוא מרצה בעירו בקורסים שונים על מחשבים ונושאים הקרובים להם. הוא מקדיש חלק ניכר מזמנו לעבודה קהילתית עם הנוער. הוא חבר בארגון נוער והיה מאמנה של קבוצת הכדורסל של ארגון הנוער הקאתולי.

על המאיירת

ילדת גרמניה שהתחנכה במאיון, מרגריט פידל היא בוגרת בית הספר לשרטוט ברוד—אייילנד. לאחר סיום לימודיה עברה מספר שנים כמחלקה האומנותית של סוכנות פירסום גדולה בניו יורק. זמן קצר אחרי שעברה לשיקאגו נתמנתה מנהלת המחלקה לאמנות בבית הוצאה של ספרי לימוד. בעיתות הפנאי נהנית מרגריט ממוסיקה קלסית ומוסיקת פופ, עוסקת בבישול לשם תחביב, אך יותר מכל אוהבת מרגריט בני אדם.

יסודות הבסיק

מבוא לתכנות בשפת BASIC

ג'ימס קוהן

בזמננו הפך הקשר בין האדם למחשב הדוק יותר ויותר. זאת, באמצעותם של מיקרומחשבים, מסופים ישירים ומרוחקים ומערכות של שיתוף-זמנים. כמו כן מספר רב של תלמידים בבתי-ספר תיכוניים, סטודנטים באוניברסיטאות, אנשי מדע, אנשי עסקים ולמעשה האוכלוסיה בכללותה, כולם מסוגלים כיום לקלוט, להבין ולהשתמש בתוכניות המחשב.

ספר זה משלב את לימוד תכנות המחשב באמצעות שפת ה"בסיק" בעזרת דוגמאות מתמטיות פשוטות.

לכל אורך הספר בולט השימוש בתרשימי זרימה. הגישה הכללית מבוססת על התקדמות מן הקל אל הכבד, מתכניות קצרות אך שלמות, לתכניות ארוכות מורכבות יותר. כל מצב אפשרי, או תשלובת של מצבים אפשריים מוצגים כך, שידגישו את הטעון הדגשה.

השימוש בפרטים נעשה בעת הצורך בלבד, ואין מטרתו להכביד על התלמיד. לשם כך מופיעות בספר בשני חלקיו 125 תכניות שונות.

כל האפשרויות הגלומות בשפת ה"בסיק" מופיעות בחלק הראשון של הספר.

בסוף הספר מופיעים נספחים אשר מספקים בין השאר הבהרות לגבי איבחון טעויות, עריכת טבלאות, דוגמאות של תרשימי זרימה. נספח אחד מביא במרוכז ובקצרה את כל סוגי ההוראות בהן משתמשים בשפת הבסיק.

בספר מספר רב של בעיות, שפתרוןן מובא בסוף הספר. דבר המאפשר לכל אחד תירגול טוב של חומר הלימוד.

מיקרו-מחשבים

בשימוש עסקים קטנים

רוברט ד. רנדול

ספר זה מהווה מעין הקדמה למשתמש במיקרומחשב.

הספר נכתב במיוחד, עבור איש העסקים - חסר הידע או בעל הידע המועט בתחום המחשבים. הספר מתאר מהו מיקרומחשב וכיצד הוא פועל. הוא מסביר פתקציות עסקיות, אשר בקלות ניתן ליישמן באמצעות המחשב. הספר מדריך בבחירת החומרה (הציוד) והתוכנה בהתאם לצרכיו המיוחדים של כל אחד ומציג דרכים לפתרון בעיות שונות באמצעות המחשב.

בספר תוכל למצוא את כל מה שאתה צריך לדעת על שימושי המיקרומחשב בעסקך. הספר יעזור לך בין אם ברשותך כבר מחשב, או באם אתה חושב על רכישת מחשב. הספר כתוב בצורה פופולרית, כך שכל אדם ללא כל ידע מוקדם יוכל להפיק את מלוא התועלת מקריאתו.

הספר כולל:

- * הסבר להבדלים בין שפות התיכנות הפופולריות השונות.
- * מתי וכיצד לבחור בחבילת תוכנה?
- * בעד ונגד של כתיבת תוכנה עצמית.
- * אזהרות ממלכודות בבחירת תוכנה.
- * כיצד לבצע עבודות יומיות, שבועיות, חודשיות ושנתיות באמצעות המחשב.
- * שיחזור וגיבוי של תכניות ונתונים.
- * ועוד, ועוד ועוד....

דף זה נותר ריק במכוון.

